Metabolomic Profiling of Nutrient Solutions for Characterization of Human Gut Microbiota

by

Cheng-Hui Sandi Yen

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Master of Science

in

Chemistry

Waterloo, Ontario, Canada, 2014

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Overview

Studying the metabolite output, the "exometabolome," of bacterial communities provides insight into the metabolic interactions occurring within that ecosystem. As such, a metabolomic approach can be used to gain further understanding of the human gut microbiota. As the microbes that reside in the human gastrointestinal (GI) system, the ecosystem has extensive impacts on the human body. *In vitro* systems for culturing gut microbes circumvents the ethical and technical challenges of *in vivo* studies, while, at the same time, lends itself to metabolomic investigations. The nutrient solutions in which microbial communities representative of the human gut microbiota are cultured contain a wealth of metabolomic data. In this thesis, metabolomic characterization is used to validate an *in vitro* gut-mimicking bioreactor system. This provided support for the ability to establish standardized protocols on preparing, culturing and reporting on *in vitro* gut microbiota. Furthermore, the metabolomic approach to studying gut microbes was used in applications relevant to clinical and industrial research. The findings from this thesis demonstrate the application of a metabolomic approach to studying the human gut microbiota in a single-stage continuous stirred tank reactor (CSTR) system. More importantly, the research here exemplifies the ways in which metabolomic insight can be translated into deciphering the complex interactions of the human gut microbiota.

Acknowledgements

This thesis would not be possible without the help and support of my mentors and peers. My thanks to Professor Marc Aucoin, who challenged me to ask the tough and relevant questions and in doing so, fostered a passion for science (and even engineering) within me. And to Dr. Emma Allen-Vercoe, who brought forth a wealth of knowledge and enthusiasm, in equal parts. I am grateful for both Marc's and Emma's mentorship throughout my degree. Thank you to my committee members, Dr. Guy Guillemette and Dr. Thorsten Diekmann for contributing time and energy into making this thesis the best that it could be.

I also extend my acknowledgements of my colleagues at the Aucoin lab. My deepest thanks go to Stan Sokolenko; thank you for your patience and your generosity. The skills you have taught me directly formed my work presented here and they will only continue to expand. Beyond this, it is impossible for me to overestimate the value of your friendship to me, as you have provided endless support to me, both in and outside of the lab. Jann Ang, Megan Logan, Steve George, Altamash Jauhar, Bhavik Manocha, Eric Blondeel, Valya Malenkov, Jian Xiong, and Biljana Todorovic, you have become my second family over the past few years. I am thankful to each of you for creating such a memorable and enjoyable workplace.

Further appreciation is extended to members of the Allen-Vercoe lab, without whom I would not have this thesis. Thank you to Julie McDonald, Erin Bolte, Kathleen Schroeter, Ian Brown and Chris Ambrose, for your time and commitment to the partnership between our two labs. As well, to Dave Chang of Chenomx Inc., who generously guided me in my introduction to the field of metabolomics. Lastly, thank you to my family who have been with me every step of the way. And to Nick, who is a constant to me.

Dedication

This thesis is dedicated to my parents, Mom and Dad, and to my siblings, Danny, Judy and Anita.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

^{13}C	Carbon
1D-^{1}H NMR	One dimensional proton nuclear magnetic resonance
1D-NOESY	One dimensional proton nuclear Overhauser effect spectroscopy
^{1}H	Proton
BCFA	Branch chain fatty acid
BHI	Brain heart infusion
CHO	Chinese hamster ovary
Clm	Clindamycin
CMV	cytomegalovirus
CQA	critical quality attributes
CSTR	Continuously Stirred Anaerobic Bioreactor
DGGE	Denaturing gradient gel electrophoresis
DPI	Days Post-Inoculation
DSS	4,4-dimethyl-4-silapentane-1-sulfonic acid
FDA	Food and Drug Administration
FOS	Fructo-oligosaccharides
FTICR-MS	Fourier transform ion cyclotron resonance mass spectrometry
GC-MS	Gas chromatography mass spectrometry
GI	Gastrointestinal
HBM	Human Baby Microbiota
HMDB	Human metabolome database
HMP	Human Metagenomic Project
HTST	High-temperature/short time
LCFA	long chain fatty acid
LC-MS	Liquid chromatography mass spectrometry
M199	Media 199
MCC	Multi-stage continuous culture
MET	Microbial ecosystem therapeutic
MetaHIT	Metagenomics of Human Intestinal Tract
MS	Mass spectrometry
MVM	minute virus of mice
NE	Norepinephrine
NIST	National Institute of Standards and Technology
NMR	Nuclear Magnetic Resonance
PC	Principal component
PCA	Principal component analysis
Ppm	Parts per million
RED	Reduction equivalent dose

rpm	revolutions per minute
SCCS	Semi-continuous culture systems
SCFA	Short chain fatty acid
UV	ultraviolet
Van	Vancomycin
VFA	volatile fatty acids

Chapter 1 Introduction

The field of metabolomics focuses on the complete set of metabolites produced by an organism, referred to as the metabolome[1]. As the summative product of an organism's biochemical processes, the metabolome produces a "snapshot" of that organism's metabolic state in that particular environment and at that particular time. A metabolomic approach could be taken to investigate the state of bacterial communities by observing the metabolite changes occurring in the nutrient solutions in which those microbes are grown. The metabolomic data of a bacterial community reveals information on both the metabolite production and nutrient consumption by the culture as a whole. As such, this type of characterization is directly applicable to understanding complex microbial consortia such as the human gut microbiota. As a consortium of bacteria residing in the large intestines of the human gastrointestinal (GI) tract, the metabolic interaction between the microbial community and the human host has direct implications on the physiology of both sides of the commensal relationship. On the one hand, there are a multitude of measurements that comment on the health and physiology of the human body, but assessment of the gut bacteria is not as straight forward. One means of studying the gut microbiota is by culturing the bacteria *in vitro* in a single-stage continuous stirred tank reactor (CSTR) system, and allowing the bacteria to grow as a community culture. There are many advantages in using such a system to study the gut microbiota, one of which is that it lends itself to metabolomic studies. In characterizing the metabolites in the spent media of the *in vitro* mixed culture, the human gut microbiota can be directly studied and subjected to experimentation.

The driving hypothesis of this thesis is that nuclear magnetic resonance (NMR)-based metabolomic data found in nutrient solution is an effective approach to studying the human gut microbiota *in vitro*. This overarching theme can be broken down into two subsequent sub-hypotheses. The first is that metabolomic data within spent media obtained from culturing a gut microbial

community can be used to evaluate the microbial changes imparted by *in vitro* growth, thereby validating the single-stage CSTR system used to mimic the human large intestines. This hypothesis will be tested by three main objectives. The first was to develop a standard method of reporting on fecal metabolomes. The second was to develop a protocol for preparing samples from the gut-mimicking bioreactor for NMR metabolomic analysis. The last objective was to determine the importance of the proportional bacterial makeup of fecal inoculum relative to the resulting *in vitro* community culture. The second hypothesis tested was that the metabolite profiles of gut bacteria communities grown in a single-stage CSTR system will sufficiently describe and compare the metabolic interactions taking place in the community culture, and define metabolic functions unique to each culture. Two main objectives were set to test this hypothesis. The first was to characterize feces-derived and defined communities that were representative of the human gut microbiota based on their metabolite signatures. The second was objective was to evaluate the communities' metabolic responses to different types of perturbations to the cultures' CSTR environment.

Establishing metabolite profiles for solutions that contained large numbers of metabolites is a difficult task with several challenges that can be overcome with training and practice. Therefore, the first goal was to demonstrate that NMR can be used to create metabolite profiles of a semi-well-defined solution such as serum-free cell culture media. In the first experimental chapter (Chapter 4), a targeted profiling approach is illustrated, followed by an experiment that demonstrates the applicability of the technique in investigating cell culture media. The experiment presented in this chapter examined the ability to create and use metabolite profiles of a chemically-defined media that underwent non-biological processing steps. More specifically, metabolite profiles were used to assess the chemical changes resulting from UV irradiation. As a nutrient solution, unspent defined cell culture media was an appropriate matrix to develop the analytical skills involved in building

metabolite profiles from complex ^{1}H NMR spectra. The examination shows an initial demonstration of targeted profiling techniques to build accurate and precise metabolite profiles.

Since fecal water has been the matrix of preference in metabolomic research on the human gut microbiota, Chapter 5 dressed the methodological concerns in reporting on fecal metabolomes. Even though fecal water is widely studied in the gut microbiota field[2–4] and is used as the inoculant for gut-mimicking CSTR systems[5,6], there is a lack of standardization of the biofluid in terms of preparation for NMR-based metabolomic analysis. Therefore, samples of fecal water of various concentrations were examined by metabolomic analysis. This systematic evaluation of fecal water samples provided evidence to suggest a means of standardizing reports on fecal metabolome.

Next, the methodological approaches for studying human gut microbiota based on metabolomic descriptions of community cultures grown *in vitro* in an anaerobic CSTR was established (Chapter 6). Two experiments were conducted to characterize different aspects of propagating mixed cultures in a CSTR that mimicked the conditions of the human colon. First, attention was directed towards developing and validating an appropriate protocol for preparing samples collected from the gut-mimicking bioreactor. To this end, bioreactor samples were prepared for NMR-based metabolomic analysis using different methods. Comparing these different preparation protocols determined the effects of syringe filtration and ultracentrifugation on the metabolite profile of a given sample. The last experiment in this chapter studied CSTR-grown cultures that started from inoculants with various proportions of bacterial biomass. Given the limited standardization of the fecal inoculants used for *in vitro* community cultures, the impact of the inoculant on the *in vitro* culture was the focus of the investigation.

The final experimental chapter (Chapter 7) applied the metabolomic analyses developed in previous chapters to characterize the human gut microbiota. The first experiment presented compared

various bacterial communities, where these communities were either feces-derived, obtained from different individuals, or obtained through the culture of a defined community that was modelled on a fecal community. The objective of this study was to describe different gut bacterial communities based on their metabolic characteristics in a single-stage CSTR system. Having established a baseline metabolomic view of the gut microbiota, the second experiment in this chapter extended our analysis to characterizing the gut microbiota under various conditions. The objective of this investigation was to capture perturbations to the cultures' metabolome when the gut microbial community was subjected to different treatments. This study was a demonstration of how metabolomic investigation could be applied in a manner that would be relevant to clinical and industrial understanding of the human gut microbiota.

Experimental work pertaining to the preparation of fecal water, fecal inoculants and defined community inoculants; development and operation of the gut mimicking CSTR system; and collection and preparation of samples were all conducted in the Allen-Vercoe lab of the University of Guelph. The focus of this thesis was on the metabolomic implications of the experimental work and to conduct metabolic analyses on the collaborative projects.

Chapter 2 Literature Review

2.1 The Metabolomic Approach

Metabolomic research, originally rooted in plant[7] and toxicology[8] research, was developed as a complementary tool to functional genomics in order to incorporate molecular interactions within our understanding of biological systems[9,10]. Since then, metabolomics has become an approach in its own right and has been applied in disease diagnosis[11], nutrition assessment[12] and microbial studies[13]. Metabolomic research is diverse in its application because it focuses on the complete set of low molecular weight metabolites produced by a given organism, referred to as the metabolome[1]. Compared to other "-omes" such as the genome, transcriptome and proteome, the metabolome is most descriptive of the organism's phenotype. The metabolome can contain a wide variety of molecules with different physical properties; from low molecular weight fatty acids and volatile organic acids to higher molecular weight amino acids, carbohydrates and lipids. Consequently, multiple analytical platforms are required to capture an entire metabolome. Furthermore, a comprehensive metabolome captures the metabolic state of the organisms at a given time, meaning that the metabolome studied is strongly influenced by temporal and spatial parameters of the experiment[14]. Therefore, a powerful metabolomic approach must begin with careful experimental design, followed by the use of analytical methods that are high-throughput and robust, and completed using appropriate statistical analyses of the collected dataset.

Given that no single analytical strategy can capture the metabolome as a whole, the choice in the platform selected involves trade-offs between advantages and disadvantages associated with each technique. There are two main analytical methods used to obtain metabolite data: mass spectrometry (MS) and nuclear magnetic resonance (NMR) spectroscopy.

2.1.1 Mass Spectrometry

Mass spectrometry (MS) is an analytical tool that identifies compounds by separating ions generated from organic or inorganic materials based on their mass-to-charge ratio[10,15]. MS is usually coupled with a separation step such as gas chromatography (GC-MS) or liquid chromatography (LC-MS), though there are several other techniques that can be employed in conjunction with MS, such as Fourier transform ion cyclotron resonance (FTICR-MS)[16]. One of the main strengths of using MS is the sensitivity of the technique, where the lower limit of detection of compounds can be on the order of picamolar or nanomolar concentrations. MS has also been demonstrated to be robust, yielding reproducible identification and quantification of metabolites[17]. On the other hand, MS-based techniques require labour-intensive preparation steps that destroy the sample in the process of extracting, separating and detecting the metabolites[18]. Intrinsic to the separation process is a bias towards which metabolites are observed, therefore requiring additional derivatization steps to broaden the scope of compounds targeted[19]. For example, GC-MS targets volatile metabolites that are also thermostable up to 280 °C, but samples can be derivatized to convert compounds from non-volatile and thermolabile to volatile and themostable before being analyzed by GC-MS[20,21].

2.1.2 Nuclear Magnetic Resonance Spectroscopy

NMR spectroscopy is another analytical tool by which to identify and quantify compounds. It functions by applying strong magnetic fields to induce atoms to spin at a precession rate proportional to the strength of the magnet applied. Within the externally applied magnetic field, about half of the nuclei will be in the lower energy “up” spin state and half in the higher energy “down” spin state. The gap between the two energy states corresponds to a specific frequency of electromagnetic radiation. A high-power radio frequency is applied to the sample to excite all the nuclei of a given type (i.e. ^{1}H). The sum of all the nuclei precessing at a specific resonant frequency form a net magnetization of the

sample. As the spins drop back from the high energy spin state to low energy spin state, the process of relaxation releases energy that can be observed by the NMR receiver as a signal, referred to as the free induction decay (FID) at that resonant frequency. Since an FID signal is a function of time, Fourier transform converts an FID signal to a function of frequency, which separates the signals into multiple peaks at their corresponding frequencies. The absorbed RF energy, which is exactly the energy gap between the two spin states, promotes nuclei in the low-energy spin-state to a high-energy spin-state. The energy emitted during the subsequent relaxation process from high-energy back to low-energy spin states is what is detected as that atom's signal. The spectrometer can use the characteristic frequency of a given nucleus, such as a proton (^{1}H) or carbon (^{13}C), and obtain information on all ^{1}H or ^{13}C atoms in a sample[4].

One of the advantages of using NMR spectroscopy is that structural information of compounds found in the sample are also relayed through the resonant frequency. An atom's structural environment impacts its resonance frequency because small magnetic fields from surrounding atoms within a molecule influences the external magnetic field in the immediate surroundings of that nucleus. These variations to the external magnetic field, localized to individual nuclei, is described as the chemical shift, and is influenced by molecular symmetry and the types of electronegative atoms nearby, which have consistent and predictable effects on its surroundings. Therefore, the chemical shift of an atom, measured in parts per million (ppm) conveys structural information about its associated molecule and is reflected based on the location of the signal along the spectrum[4]. Additionally, nuclei in close proximity (within three bonds) affect the external magnetic field experienced by neighbouring nuclei, an effect called spin-spin splitting. Therefore, neighbouring nuclei mutually alter each other's resonance frequencies and alter the magnetic field experienced. Spin-spin splitting is represented in the spectrum by "splitting" the resonance peak corresponding to

the nucleus of interest into multiple peaks, set apart by J Hz, where J is the coupling constant. The pattern of peak splitting depends on the number of neighbouring nuclei and the intensity of the split peaks is determined by Pascal's triangle[4].

NMR has also been demonstrated to be a robust analytical tool that yields reproducible results suitable for metabolomic analysis of biofluids[22,23]. Furthermore, the sample preparation process for NMR spectroscopy is non-destructive and simple as the preparation procedure only involves adding an internal standard compound in deuterated solvent to the sample to control for drift in the magnetic field and for downstream quantification. Another benefit of using NMR spectroscopy for obtaining metabolomic data is the non-selective nature of the technology. Since NMR utilizes the quantum properties of molecules, the resonant frequency of all atoms are affected by the external magnetic field. The NMR spectrometer can target the characteristic frequency of a particular type of nucleus to obtain the resonance frequency information of all metabolites that contain the targeted atom within the sample. Most applications of NMR utilize ^{1}H or ^{13}C atoms to relay the metabolite make-up of a sample. This is because the sensitivity of NMR spectroscopy is directly related to the abundance of the targeted atom as well as the strength of that atom's nuclear magnet. For example, the reduced sensitivity of NMR to ^{13}C signals in comparison to ^{1}H is compounded by the fact that ^{13}C exists at much lower natural abundance (about 1.1%) and has a weaker nuclear magnet than ^{1}H. Since the sensitivity of the platform is dependent on the abundance of nuclei, NMR spectroscopy has relatively weak sensitivity. While there are hardware solutions to improve signal intensity and sensitivity, the lower limit of detection using NMR spectroscopy is still on the order of micromolar concentrations[24].

Just as there are many different throughput and separation techniques used to enhance various attributes of MS, there are many different NMR experiments that adjust the spectral acquisition

process to augment particular characteristics of NMR as a platform for obtaining metabolomic data. For example, there are experiments that aid with water suppression[25,26], improve signal de-convolution[27,28], or high-throughput interpretation of NMR spectra[29,30].

1D-^{1}H-nuclear Overhauser effect spectroscopy (1D-NOESY) pulse sequence has become the NMR experiment of choice when it comes to performing NMR-based metabolomic studies[23]. Inherent to its popularity, one of the main benefits of using 1D-NOESY is that, as an established procedure, the results obtained from the pulse sequence can be consistently compared with other studies using the same experiment. Moreover, the widespread use of ID-NOESY in metabolomic research has led to development of downstream processing tools, such as Chenomx NMR Suite (Chenomx Inc., AB), that facilitate the interpretation of the acquired spectra[31]. One factor that likely contributed to the ubiquitous use of ID-NOESY for metabolomic analyses is the compatibility of most modern NMR spectrometers with the hardware requirements of the experiment. In terms of the spectroscopic advantages of the ID-NOESY, it is an experiment that requires minimal optimization while still conferring adequate suppression of solvent signals[32]. 1D-NOESY exploits the difference in T_1 relaxation time of the solute and solvent in order to discriminate the two components of the matrix. Another aspect of 1D-NOESY that contributes to the successful suppression of the solvent signal is by using a series of pulse sequences to select against resonances from parts of the sample that did not experience the pulses at full efficacy or experienced non-homogenous fields. The third mechanism of 1D-NOESY that facilitates selection against unwanted signals is through the use of phase cycling. Phase cycling is a complex mechanism as its effects are embedded within many aspects of the experiment. Generally, signals of interest are “cycled through” various phases, so it is reversed relative to the phase of the NMR receiver. This way, external noise source contaminating a signal are subtracted while the signal of interest is added when alternately reversing the direction of the desired

signal and the NMR receiver. Given the quality of solvent suppression and the ease of integrating the resultant spectra into downstream analyses, this experiment can be appropriately applied to metabolomic studies.

2.1.3 Principal Component Analysis

The rich information contained within complex metabolomic data can be extracted using appropriate statistical tools. Multivariate analyses have been applied to interpret quantitative and qualitative annotations of metabolites, such as compound concentration and compound identity[33,34]. Principal component analysis (PCA) is one such analysis that provides an overall view of the data structure, revealing trends and groupings within the data. This pattern recognition technique allows visualization of datasets that have many correlated variables associated with each observation. Within metabolomics, the metabolites are the variables associated with each observation, and an observation is a metabolomic spectra (NMR or otherwise). PCA is able to create a model based on the samples and their respective metabolite profiles by transforming the correlated variables into a reduced set of new, independent variables, called principal components (PCs)[35]. Mathematically, the original data is represented as the covariance measured between every variable to form a covariance matrix. Once the variable relationships are captured in terms of their covariance, the eigenvectors of the covariance matrix represent score coordinates and the eigenvalues of the covariance matrix represent the loading values. The interpretation of these new PC variables in the context of metabolomics is that the score coordinate values illustrate how the spectra are related to one another, and the loading values depict how the metabolites contributed to the PCs. Graphically, when spectra are located along a given PC, it demonstrates that they are related along that axis, while the coordinate value of a metabolite in the same PC space illustrates how that metabolite relates to those spectra. Loading plots illustrate how the metabolites relate to those spectra[36]. The ability to visually detect patterns of similarity and

differences within metabolite profiles is the main strength of PCA and has been the driving force for its popularity in metabolomic analysis.

2.2 ^{1}H NMR-Based Metabolomics

As mentioned earlier, the NMR spectrometer can use the characteristic frequency of different nuclei to relay information on the structure and concentration of metabolites in a sample. Since the sensitivity of the NMR is directly related to the abundance of that atom, monitoring the ^{1}H provides the greatest relative sensitivity as it is found at almost 100% natural abundance. For this reason, one dimensional proton NMR (1D-^{1}H NMR) is often employed for obtaining metabolomic data in various fields, including cell culture research[37] and study of human biofluids[38]. 1D-^{1}H NMR spectra can provide quantitative identification of the spectral peaks through "targeted profiling," a method whereby spectral peaks are individually matched with metabolite signatures obtained from a library of pure compound spectra[31]. This method of analyzing the spectra allows one to interpret the spectral signals, even when the peaks overlap, a characteristic often encountered in 1D-^{1}H NMR spectra of complex solutions. Previous assessment of targeted profiling determined that the accuracy and precision of quantification is influenced by the combination of the degree of convolution affecting the metabolite's signals and the magnitude of the compound's concentration; compounds that are highly convoluted and found at low concentrations will have greater measurement error[39]. Since measurement error in targeted profiling approaches are compound-dependent, peak-specific strategies have been developed to improve measurement accuracy[39–41]. These strategies involve group-wide review of profiled spectra to establish confidence in each metabolite's observation; generation of a master profiling list for consistent metabolite identification; or performing a validation experiment to define quantification accuracy within that sample type.

2.2.1 Metabolomic study of Cell Culture Processes

Within the framework of a single-cell organisms, the metabolome is composed of the endometabolome and the exometabolome, which refers the organism's intracellular metabolites and extracellular metabolites respectively[13]. Since the metabolome is regarded as the most accurate description of the organism's phenotype, it can be applied as an analytical tool in cell culture processes. For example, metabolomics has been used in culturing various cell lines, providing complementary data to genomic and proteomic measurements. In the field of industrial biotechnology, areas of focus for cell line metabolomics include phenotype classification, monitoring bioprocesses and ensuring consistent product quality. For example, metabolomic data have been used as a high-throughput screening tool for strain classification. *E. coli* strains containing mutations in the tryptophan metabolism pathway were identified based on the unique metabolite profiles associated with altered tryptophan pathways[42]. Discriminating mutant strains using metabolite profiles is directly relevant to industrial biotechnology since screening for and identifying cell line variants that strongly impact downstream products is highly desirable. Cell line metabolomics is also relevant to industrial processes as it can contribute to monitoring cell physiology during production. One example is in production of fuel ethanol using *Saccharomyces cerevisiae*. The environments imposed on the yeast cells during fermentation can illicit stress responses that could ultimately impact the end product[43]. In such a case, the metabolic changes undergone by the cells in response to the harsh environments of fuel ethanol production were identified and contributed to determining the ideal growth conditions and media supplementation required for particular strains of yeast[44]. Using metabolomic data for bioprocess monitoring also allows one to assess the production process. For example, NMR-based metabolomics was used to assess the scalability of producing a therapeutic protein in Chinese hamster ovary (CHO) cells[45]. Aranibar et al found that biochemical signatures associated with differences in

production levels in a 5000-L and 7-L bioreactors were related to energy metabolism, like adenosine nucleoside phosphates and UDP-glucose, 1,6-glucose bisphosphate and UDP-galactose. Another area of consideration in production of bio-therapeutics is assessing the nutrient feed provided to the cell cultures. Referred to as "fermentanomics," assessing metabolite components in media[46,47] could reveal information on product quality and cell culture productivity. Nutrient consumption and metabolite accumulation in the media can strongly influence cell culture performance[47,48]. Therefore, rational medium designs paired with NMR-based monitoring have been executed to direct nutrient consumption in order to control product quality[46]. One such example of this was the marked increase in the production of monoclonal antibodies in CHO cells by supplementing the media with additional methionine, tryptophan and tyrosine due depletion of those amino acids in conditions of excess glutamine and glucose. [47]. The wide-ranging applications to which metabolomic data contributes to cell culture processes attests to the depth and versatility of the tool. Beyond analysing pure cell cultures, which provides metabolomic views of an entire organism, the metabolomic approach also has potential in deciphering whole biological systems, such as the human body.

2.3 The Human Gut Microbiota

The microbiota of the human gut is an intricate community that interacts symbiotically with the human host to influence health and physiology. It is estimated that the number of microorganisms residing within the human gastrointestinal (GI) system (Figure 2-1) is approximately 10^{14} cells, ten times that of the number of human cells in the body[49]. Such a complex ecosystem of microorganisms, mostly bacteria[50], interact closely with its environment via genetic and metabolic signals. Over the past decade, there has been an increased interest in characterizing the human gut microbiota through analyses of such signals. On one hand, the establishment of the Human Metagenomic Project (HMP) Consortium and the Metagenomics of Human Intestinal Tract (MetaHIT) reflect some of the

collective efforts researchers have put forth to use genetic characterizations to identify the microbiota and the roles they play within the microbial ecosystem and with the human host[51,52]. On the other hand, metabolomic studies have been used to explore and understand relationships involving the gut microbiota which are not encoded within the genome[53]. Complementary to metagenomic studies, researching the metabolic interactions amongst the gut microbiota and with the host cells have contributed tremendously to establishing relationships between bacterial communities and with the human body, in healthy and disordered states[54].

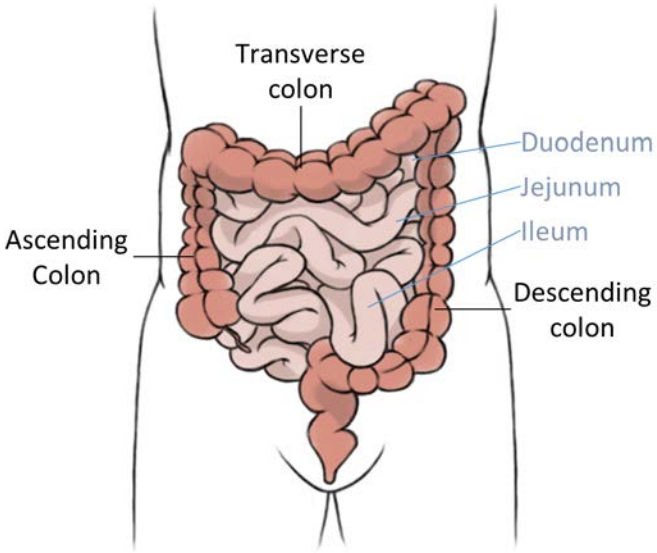

Figure 2-1. Anatomy of the human GI tract. Segments of large intestines labeled in black; segments of small intestines labelled in grey.

2.4 Metabolomic Study of the Human Gut Microbiota

When applied to bacterial communities, metabolomic approaches give insight into the metabolic activities engaged by the entire population. Within the context of the human gut microbiota, the mixed culture metabolome not only reflects the metabolic activities of the enteric bacteria, but it also represents the bacterial compounds to which the human body is exposed. Metabolomic studies of the human gut microbiota have been heavily focused on analyzing the compounds found in feces. Understanding gut bacteria through feces is an attractive approach as the procedure is non-invasive, yet still provides a wealth of information on the human-gut microbe relationship. Specifically, the study of the aqueous phase of feces, referred to as fecal water, has provided valuable information regarding host-bacterial changes under different conditions. The fecal metabolome was demonstrated to be an effective tool in assessing dietary effects, such as polyphenol-rich diets[55], daily apple intake[56], and red meat consumption[57]. In clinical research, investigating the fecal metabolome has resulted in biomarkers for Crohn's disease[58] and colorectal cancer[59], and has been an effective diagnostic tool for ulcerative coslitis[3] and irritable bowel syndrome[60]. The ability to obtain these findings via a non-invasive procedure is one of the main benefits of using feces to study the gut microbiota. Unfortunately, fecal samples have been shown to vary greatly between individuals and within an individual over time due to differences in gut physiology and variable diets, which in turn can affect properties of fecal samples, such as pH, moisture level and metabolite concentrations[61]. Therefore, one of the main drawbacks of examining stool samples is the heterogeneity of the matrix and the lack of standardization in preparing and analyzing the samples. There has been some effort in establishing optimised preparation procedures of fecal samples for the purposes of analyzing just the exometabolome[62], or both the intra- and exometabolomes[55]. However metabolomic data obtained from different preparation protocols and different batches of feces still cannot be performed without a

means of standardizing the sample's content. Interpretation of fecal metabolomes is further complicated by the fact that it is a reflection of host-microbe co-metabolism, where the fecal metabolome represents the cumulative effects on the host and the gut microbiota. Reproducibility of the metabolomic data is another obstacle in fecal metabolome research. The exact composition of the gut microbiota has been shown to vary with host age, nutrition, behaviour and health[2,54,63], so controlling for such experimental factors or recreating the exact conditions of previous studies can be quite difficult. Therefore, a system that allows for control over the experimental parameters subjected to the gut microbiota would circumvent the challenges of using fecal samples to study the gut microbiota.

2.5 Studying the Gut Microbiota using Model Organisms

The use of model organisms such as mice and rats allow for greater control over the subjects' diet and environment. The use of model organisms also widens the scope of experimental conditions that could be researched when it comes to understanding gut-host interactions, as some treatment conditions, such as antibiotic exposure, would have ethical limitations for human studies[64]. For example, the administration of imipenem/cilastatin sodium, a broad spectrum β-lactam antibiotic to Wistar rats housed in a specific-pathogen-free environment was performed to assess its effects on the gut microbial-mammalian co-metabolism[63]. Examination of the metabolome of both urine and feces demonstrated that profound changes to the host and microbe metabolism occurred rapidly (within the first day of treatment), while recovery from antibiotic treatment was gradual. The main metabolites that underwent large fold changes with treatment included aromatic benzene compounds, carbohydrates, short chain fatty acids (SCFA) and compounds of the tryptophan metabolism pathway. The metabolic perturbations observed were attributed to impacts on the gut microbiota rather than to host metabolic pathways since the antibiotic used had poor systemic absorption. At the same time,

host-microbial co-metabolism was still observed as there were changes to metabolites that are found in both mammalian and microbial physiology. Given that the gut microbiota are different across different species of model organisms and from humans[2], the applicability of studies performed *in vivo* can be enhanced through "humanizing" the model organisms' gut microbiota. Germ-free or gnotobiotic mice, which are devoid of or have defined gut microbiota, can be inoculated with microbial consortia derived from human donors. Using this model, Martin et al. introduced human baby microbiota (HBM) to germ-free mice and examined the effects introducing *Lactobacillus paracasei* as a probiotic and a two dietary regiments (glucose-lactose mix and a galacto-oligosaccharaide-rich prebiotic mix); all the while comparing them to conventional and conventionalized mice[54,65]. Generally, they found that HBM mice had different levels of bacteria compared to the HBM mice that were supplemented with a probiotic, which also had different levels of bacteria than conventional and conventionalized mice. On the other hand HBM mice given a probiotic had similar levels of gut microbes as HBM mice supplemented with both a prebiotic diet and probiotic treatment, though the HBM mice subjected to the prebiotic diet had greater levels of bifidobacteria over time. In examining the fecal metabolic profiles, specific metabolic features could be attributed to the HBM community, the probiotic treatment and the two dietary regiments (basal glucose-lactose mix and prebiotic feed). Many of these identifying metabolic signatures involved changes to SCFA levels, such as acetate, butyrate and propionate, amino acid changes and also compounds associated with energy metabolism, such as succinate, glucose and lactose. Furthermore, there was a time-scale component to the changes observed, where the metabolite profiles of conventional and conventionalized mice were stable over time, while the HBM mice under different diets diverged from each other over time. These experiments conducted by Martin et al. exercised the power of NMR-based fecal metabolomics to examine the human gut microbiota under modulated environmental conditions. As exemplified in the studies described, studying the fecal metabolome in

an *in vivo* model allows one to assess the host-microbial co-metabolism and retain a moderate level of control over the experimental parameters, such as diet, environmental exposure and pharmaceutical treatment.

2.6 Studying the Gut Microbiota *In Vitro*

2.6.1 Batch Fermenters

The gut microbiota can also be studied through *in vitro* models. Where there are many model designs, the basis of these *in vitro* models involve the growth of fecal microbiota in a fermentation reactor operating anaerobically under physiological conditions. As the complexity of the bioreactor designs increase, so does the resemblance of the model to biological conditions. Batch reactors consist of culturing fecal inoculum in a single vessel supplied with nutrients, though without replenishment. Since batch reactors grow the bacteria in a closed system, the growth medium strongly influence the growth of the community culture. In one experiment[6], the microbial communities reached similar maximum cell densities at the same time (18 h) in three kinds of brain heart infusion (BHI) media that varied in carbohydrate concentration. The different kinds of media had varying ability to maintain the initial fecal microbiota as measured by denaturing gradient gel electrophoresis (DGGE). For example, the low-carbohydrate BHI medium had the greatest ability to maintain the inoculum culture as indicated by similar DGGE profiles, while the high-carbohydrate BHI medium resulted in a DGGE profile with considerably fewer bands. In general, the *Bacteroidetes* phylum was more dominant in *in vitro* cultures, whereas the *Firmicutes* phylum were less dominant in *in vitro* cultures. Batch reactors are relatively cost effective, and easy to operate[66]. For these reasons, batch reactors are suitable for experiments geared towards high-throughput screening for specific metabolic functions[6], such as bioconversion of plant lignans by gut microbes[67]. Batch fermenters accommodate growth of fecal

microbes only on short-term basis (24-48 h) because extended growth of the complex community can lead to unwanted selection of certain bacterial strains as nutrients are quickly consumed and the environment in the batch system changes over time. For this reason, continuous reactors that allow a more constant environment by replenishing nutrients and controlling pH are favored to mimic natural physiological conditions.

2.6.2 Single-Stage Continuous Culture Systems

Single-stage continuous stirred tank reactor (CSTR) models or single-stage semi-continuous culture systems (SCCS) allow for sustained nutrient feeding to the fecal microbes, and permit longer culture durations on the scale of several weeks. CSTR and SCCS models have been validated to maintain metabolic activity close to that of the initial fecal inoculum, as measured by hydrolytic and reductive enzyme activities, production of volatile fatty acids, hydrogen, methane and carbon dioxide[68]. Shifts in fecal bacteria being cultured in single stage CSTRs and SCCSs include the enrichment of bacteria in the phylum *Bacteroidetes* and diminishment of bacteria in the phylum *Firmicutes*, similar to changes observed in communities grown in batch reactors. These shifts were independently observed in several validation studies, though the extent of the population shifts differed across studies. For example, Allison et al[69] observed slight decreases in overall count of anaerobes, with slight increases in *Bacteroidetes* and decreases in *Firmicutes* genera like *Clostridia*, and *Eubacteria*, though increases in other *Firmicutes* like *Lactobacilli* and *Streptococci* were also observed over a 14-day period. Campbell et al[68] saw a decrease in the total number of microbes in their SCCS-cultured fecal communities. The fecal inocula contained seven different genera, with 81% being *Bacteroides* and *Eubacterium.* 31% of anaerobic colonies could not be subcultured. In comparison, the 21-day *in vitro* culture only had four different genera characterized, with 89% being *Bacteroides, Eubacterium,* and *Clostridium* spp and only 13% of isolated anaerobes could not be subcultured. In the CSTR system

used by McDonald et al[5], fecal microbial communities became enriched in phylum *Bacteroidetes* and diminished in phylum *Firmicutes*, such as *Clostridium* (cluster XIVa), and *Bacilli* after 4-5 weeks in the *in vitro* model.

Single stage CSTR systems and SCCS reactors also lend themselves to studying the effects of prebiotics and probiotics on gut microbial communities. Prebiotics are nondigestible food ingredients that selectively stimulate the growth or activity of bacteria in the GI tract, with the purpose of beneficially impacting the host[70]. Probiotics refers to live microbial supplements used to improve the intestinal microbial balance, with the aim of beneficially impacting the host[71]. Duncan et al[72] used a range of carbohydrates to study substrate impact on microbial composition using 16S rRNA probes. The investigators then continued on to use these substrates to screen for highly competitive strains that had potential to be used as new probiotic strains. The total number of anaerobes remained the same with changing substrates, except when fed with pectin, which causes a global decrease in the number of anaerobes. Furthermore, specific bacterial populations were strongly impacted by certain substrates. Dhalia inulin increased *Eubacterium cylindroides* by 100-fold and *Ruminocoocus bromii* and *Ruminococcus flavefaciens* by 10-fold. The change in bacterial composition was also reflected in the production of SCFA. The proportional make-up of acetate-propionate-butyrate metabolites showed that acetate production was greatest with pectin (75%-16%-8%) and that growth on amylopectin produced the highest proportion of propionate and butyrate (44%-42%-14%). The impact of different substrates on the ability of probiotic candidates to compete with resident gut microbiota varied, with inulin and amylopectin facilitating the survival of *Rosuria* strains. Another application of single-stage CSTR has involved the study of fructo-oligosaccharides (FOS) as a prebiotic to prevent the degradation of isoflavone genistein by the gut microbiome and preserve its anti-carcinogenic properties[73]. Steer et al. found that FOS increased the total count of anaerobic

bacteria, where *Bifidobacterium* spp. and *Lactobacillus* spp. were enriched and *Bacteroides* spp. and *Clostridum* spp. were decreased. Additionally, isoflavone genistein degradation was detected to a lesser extent with FOS-supplemented media than the unmodified recipe. These two examples illustrate the applicability of using single-stage continuous reactor systems to investigate the metabolic functions of the gut microbiota beyond the screening capabilities of batch reactors.

2.6.3 Multi-Stage Continuous Culture Systems

Characterizing the entire large intestine as a continuous culture system is still a simplification of the organ's operation. In truth, only the ascending colon receives steady nutrients from the small intestines and performs peristaltic mixing of the colon contents. The transverse and descending colon experience restricted nutrient supply and no mixing. As such, the different sections of the colon – ascending, transverse and descending – can be represented *in vitro* by using multi-stage continuous culture (MCC) systems. There have been many types of reactor designs that aim to recreate the spatial, temporal, and biochemical aspects of the large intestine. Bearne et al. designed a two-stage reactor system, where the two vessels differed in pH[74], while Macfarlane and Gibson developed a three-stage continuous culture system. In the three-stage system, the ascending, transverse and descending colon are mimicked by connecting three vessels differing in pH, operating volumes and dilution rates[75]. Compartmentalizing the different segments of the large intestines revealed that the bacteriological content of the three reactor resembled that of the large intestines in terms of total anaerobic counts, but differed from *in vivo* examples in terms of species make up. Population densities of *Bifidobacteria* and *Bacteroides* were similar to the human large intestines. But the levels of *Clostridium perfringens* were greater in the first and second reactors than the ascending and transverse colon, while *Escherichia coli* was greater in the third reactor than in the descending colon. These species variation from one reactor to the next also meant that the vessels were also different

metabolically. Carbohydrate fermentation occurred mostly in the first “ascending colon” reactor, while an increase in branch chain fatty acids (BCFA) and phenolic compound due to the fermentation of amino acids that was observed in the second “transverse colon” reactor and third “descending colon” reactor.

A more comprehensive representation of the human gastrointestinal (GI) system is the Simulator of the Human Intestinal Microbial Ecosystem (SHIME) system developed by Molly et al[76]. In the SHIME system, five consecutive reactors varying in working volume and residence time are used simulate the entire gastrointestinal tract. The first two reactors mimic the small intestine, where the duodenum and jejunum are represented in the first reactor and the ileum is represented in the second reactor. These reactors received successive inoculations of nutrition suspension representative of human western diets. The last three reactors represented the three sections of the large intestine with each vessel receiving a single fecal inoculation. Enumeration of the bacteria revealed that the microbial compositions in the SHIME system were comparable to *in vivo* communities. Metabolic assessments were made through measurement of volatile fatty acids (VFA), headspace gases (CO_2, CH_4, and H_2) and ammonium. Based on these parameters, there were significant differences between the SHIME culture and *in vivo* values from literature. Acetic acid, and propionic acid were generally found at higher molar ratios and butyric acid was consistently detected at lower molar ratios in the SHIME simulator compared to *in vivo* literature values. The headspace gas composition and ammonia levels were consistent across the various media types used, but differed from reference values. This divergence could be attributed to the use of O_2-free N_2 to maintain an anaerobic atmosphere. Constantly sparging the headspace with N_2 has been shown to impact bacterial metabolism[77]. Recently, there have been modifications made to existing single stage and MCC systems to create anaerobic conditions using only the gasses produced by the fecal bacteria[78,79].

2.6.4 Mimicking Physiological Characteristics

The continuous culture systems described thus far allow metabolites and microbes to accumulate in the reactors until steady-state is reached all the while having both compounds and microorganisms removed together. The drawback of this system is that *in vivo,* metabolite concentrations are maintained relatively low as they are absorbed by the host. The accumulation of metabolites in *in vitro* systems can inhibit growth of certain microbes. One design that has attempted to address these concerns, has incorporated peristaltic mixing, water absorption and metabolite absorption[80,81]. In this design, a dialysis membrane enclosed in a tubular system has been used to remove bacterial metabolites and water. This, combined with flexible silicone membranes undulated by water pressure to simulate peristaltic mixing of luminal contents, further allows the reactor to be inoculated at high-densities despite the viscosity of the solutions. In this system, microbial stabilization was reached in 24 hours and the *in vitro* community was sustained stably for over three weeks. Validation and characterization of *in vitro* microbial diversity in this system[80,82] has led to applications such as investigating the effects of various carbohydrates, such as inulin[83] and lactulose[84], on microbial composition and metabolic activity and attributing carbohydrate fermentation functions to specific microbial populations by using isotope-labelled substrates and isotope-labelled RNA probing[85,86].

2.6.5 Challenges of *In Vitro* Models

Despite the advances in developing bioreactor designs to mimic the human GI tract, there are still challenges and limitations to using *in vitro* models. With respects to physiological factors, *in vitro* studies of gut microbiota lack host-microbe interactions. As such, the impact of the microbial activities on the host organism and the host's feedback reaction to the microbes cannot be measured. Furthermore, direct comparisons of community cultures from different fermentation experiments, even those using similar models, cannot be made. Part of the variability is attributed to inter-

individual differences in the bacterial make up within the gut. Since the gut microbial ecosystem is individually unique, *in vitro* communities are inherently different as they are derived from feces collected from different sources. Therefore, inter-individual variability must be taken into account in experimental designs, ensuring that treatment or disease effects are greater than biological variability. In terms of engineering challenges, variable retention times in the bioreactors is another confounding factor that impairs cross-experimental comparisons. Child et al[87] found that retention time strongly influences substrate availability, and bacterial colonization, which in turn shapes the entire *in vitro* community. Some bioreactor model validation studies take this factor into account and characterize its effects[75,81]. Another aspect of consideration is defining when steady-state has been established. The amount of time required to reach steady-state is highly variable, even within a single bioreactor design. For example, single-stage continuous culture systems have been observed to reach steady-state as early as 5 days to as late as 4-5 weeks after inoculation[5,68,69]. Since the achievement of steady-state is not independent of retention time, it is expected that the time required to reach a stable microbial environment is variable across *in vitro* models. Rather, attention should be focused on establishing a definition of steady state, whether it be based on microbial composition, metabolic production, or functional activity, and confirm its achievement within a given model. Lastly, *in vitro* models could easily lend themselves to metabolomic studies. However, the large majority of the *in vitro*-based investigations have placed its emphasis on microbial composition and microbial activity. Considerations of metabolic functions were usually probed only to the extent of VFAs of SCFAs or gaseous products such as H_2, CO_2, CH_4 and NH_4. Holistic examination of the metabolome of *in vitro*-propagated communities could be the next avenue of characterization in the movement to understand the human gut microbiota.

Chapter 3 Materials and Methods

3.1 NMR Sample Preparation

Samples analyzed with NMR spectroscopy were prepared to contain 10% internal standard solution, which was composed of 5 mM 4,4-dimentyl-4-sialpentane-1-sulfonic acid (DSS) and 0.2% w/v sodium azide in 99.9% D_2O (Chenomx Incs., Edmonton, Canada). This internal standard solution served as a chemical shape indicator (CSI) for software analysis of the spectra, while at the same time inhibiting bacterial contamination of the samples. The sample-DSS solutions were vortexed and transferred to 5-mm NMR tubes (NE-UL5-7, New Era Enterprises Inc., Vineland, NJ) for scanning. Samples were prepared on the day of scanning, and kept on ice until ~10 min before being scanned to allow the sample to return to room temperature.

3.2 NMR Spectra Acquisition and Processing

The 1D-^{1}H NMR spectra were acquired at the University of Waterloo Nuclear Magnetic Resonance Facility using a using the first increment of a NOESY pulse sequence (1D-NOESY) on a 600 MHz Bruker Avance NMR spectrometer. The NMR spectrometer was operated using XWIN-NMR 3.5 software and a Triple Resonance Probe (TXI 600). The 1D-NOESY pulse sequence compatible with the Chenomx software used to analyze the NMR spectra involved a 1 s presaturation that consisted of a 10 ms relaxation delay with a 990 ms presaturation period for water suppression at an induced field strength/gammaB_1 of ~80 Hz[41]. This was followed by two 90° high power pulses and a 100 ms mixing time and another 90° high power pulse before a 4 s acquisition period. The total recycle delay in the experiment was 5 s. A total of 4 steady state scans and 32 scans were performed to obtain NMR spectra for a given sample (Figure 3-1).

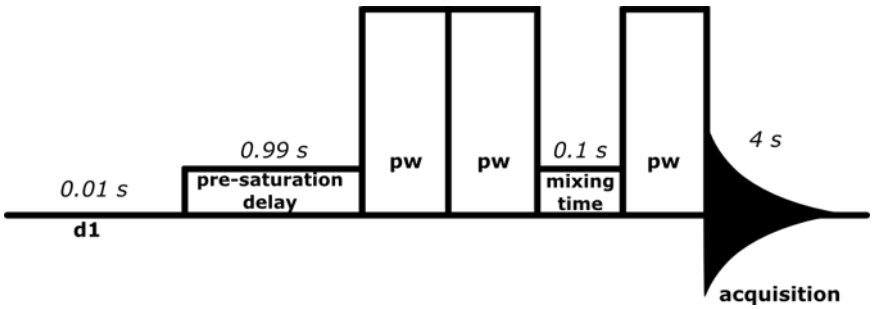

Figure 3-1. Schematic of 1D-NOESY pulse sequence. Figure credited to Chenomx application note on Investigating the criteria for accurate quantitative results with Chenomx NMR Suite[88].

Chenomx software was used to construct metabolite profiles from the 1D-^{1}H NMR spectrum (Figure 3-2). The corrected experimental spectra were assigned compounds by targeted profiling, superimposing compound signals in the built-in library to spectral peaks. The area of the spectrum signal relative to the reference compound was used for metabolite quantification. Profiled concentrations were divided by 0.9 to adjust for dilution with the internal standard solution during sample preparation for NMR spectroscopy. Sample spectra in each experiment were profiled according to scan order.

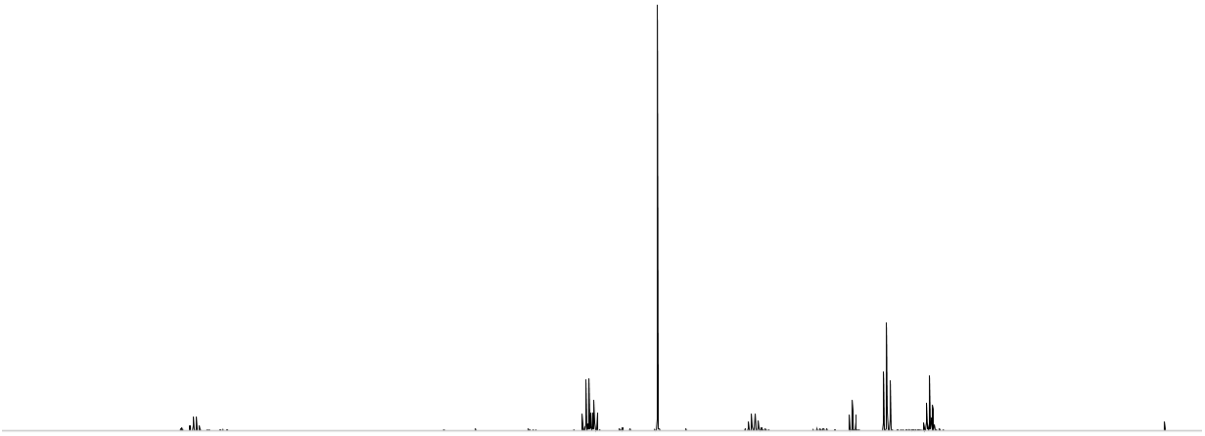

Figure 3-2. NMR spectrum of typical gut-mimicking single-stage CSTR sample after phase, baseline and shim correction steps.

3.3 Metabolite Profiles

The Chenomx software was also used to construct metabolite profiles from the 1D-^{1}H NMR spectrum. The corrected experimental spectra were assigned compounds by targeted profiling, superimposing compound signals in the built-in library to spectral peaks allowed for assignment of compound identity, and the area of the spectrum signal relative to the reference compound was used for metabolite quantification. Sample spectra in each experiment were profiled according to scan order. Once the metabolite profiles were assembled, each compound was assessed post hoc for confidence in the compounds' identification and quantification. Confidence in a compound was determined by the number of clusters in the compound's NMR signal, level of convolution with other compound signatures, and the signal-to-noise ratio of the compound. Only high-confidence compounds were included in subsequent analyses.

3.4 Statistical Analysis

All statistical analysis was performed using R, the statistical programming language[89]. The following packages were used: RSQLite, RODBC, ggplot2, gridExtra, reshape2, plyr, and RColorBrewer and all base packages accompanying R.

3.5 Bioreactor Operation and Inoculation

Bioreactor operation and experimentation was performed by members of the Emma Allen-Vercoe lab at the University of Guelph. Preparation and operation of the single stage, continuous stirred-tank reactor has been previously described[90,91]. The Infors Multifors bioreactor system (Infors, Switzerland) was used throughout this body of work. 500-mL Multifors bioreactors (3 Multifors systems, Infors AG, Switzerland), with working volumes of 400 mL were operated as continuous

stirred tank reactors. Conditions were set to mimic the distal human gut (37ºC, pH 6.9-7.0, gentle agitation, oxygen-free, and fed with a constant supply of mucin and insoluble starch substrates at a flow rate of ~400 mL/day). Maintenance of pH was automated with addition of 5% v/v HCl and 5% w/v NaOH. Vessels were maintained anaerobic by bubbling O_2-free N_2. The growth medium used in the bioreactor system was developed by the Emma Allen-Vercoe lab at the University of Guelph, and has been previously described[90,91], and is outlined in Table 3-1. Prior to inoculation, a sample of growth medium was collected from the bioreactor vessel to assess for bacterial contamination. The media sample was plated on fastidious anaerobe agar (Acumedia; Lansing, Michigan) supplemented with 5% defibrinated sheep's blood (Hemostat Laboratories; Dixon, California) and incubated in aerobic or anaerobic conditions, at 37 °C. The CSTR system used to culture the mixed bacterial communities was inoculated with either 25% (v/v) of 10% (w/v) fresh fecal slurry, or 12.5% (v/v) of defined culture suspension relative to the bioreactor working volume. Once inoculated, the culture was gently agitated. Medium feed was started 24 h post-inoculation.

Table 3-1. Composition of growth medium.

Reagent[1]	**Concentration (g/L)**
Peptone water[b]	2
Yeast extract[c]	2
$NaHCO_3$[b]	2
$CaCl_2$[a]	0.01
Pectin (from citrus)[a]	2
Xylan (from beechwood)[a]	2
Arabinogalactan[a]	2
Starch (from wheat, unmodified)[a]	5
Casein[d]	3
Inulin (from Dahlia tubers)[d]	1
NaCl[a]	0.1
$K_2H_2PO_4$[a]	0.04
KH_2PO_4[b]	0.04
$MgSO_4$[e]	0.01
Bile salts[a]	0.5
Hemin[e]	0.005
Menadione[a]	0.001
L-cysteine HCl[b]	0.5
Porcine gastric mucin (type II)[a]	4

[1]Chemical suppliers are indicated in superscripts: (a) Sigma-Aldrich (Oakville, Ontario); (b) Thermo Fisher Scientific (Ottawa, Ontario); (c) BD (Franklin Lakes, New Jersey); (d) Alfa Aesar (Ward Hill, Massachusetts); (e) BDH (Radnor, Pennsylvania). Chemical suppliers Adapted from McDonald et al, 2014[5].

3.6 Collection and Preparation of Fecal Inocula

Collection of fecal samples and the preparation of fecal inocula was performed by members of the Allen-Vercoe group as per the approval of the Research Ethics Board of the University of Guelph (REB#09AP011). The procedure was previously described and is summarized here[90,91]. Fresh fecal samples were obtained from four healthy individuals (Table 3-2), none of whom had a recent history of antibiotic treatment prior to time of donation. Donors provided fresh fecal samples in a sealed sterile plastic container which was placed in an anaerobic chamber (atmosphere 90% N_2, 5% CO_2 and 5% H_2) within 5-10 min of defecation. Each fecal samples were processed into a 10% (w/v) fecal slurry by mixing 5 g of feces in 50 mL of growth medium for 1 minute with a stomacher (Tekmar Stomacher Lab Blender, Seward; Worthing, West Sussex, UK). Once homogenized, the fecal slurry was centrifuged (10 min, 175 x g) to remove large particulates, such as residual undigested food[92]. The remaining supernatant was used as fecal inoculum for the gut-mimicking bioreactors used in the studies.

Table 3-2. Fecal sample donors.

Donor	Sex	Age (yr)	Time elapsed between most recent antibiotic treatment and sample collection
A	Male	44	6 years
B	Female	26	9 months
C	Male	25	11 months
D	Female	42	8 years
E	Female	7.72	6 months

3.7 Preparation of Defined Communities

The defined community cultures were prepared and described by Schroeter, 2014[91]. These defined bacterial communities were referred to as Microbial Ecosystem Therapeutic (MET) cultures as they were originally developed as a community culture probiotic for therapeutic purposes. In short, pure bacterial isolates, identified by 16S rRNA gene sequencing, were obtained from fresh feces collected from two different donors: Donor D, a 41-year old female and Donor A, a healthy 44-year old male. 33 species were selected from the axenic fecal cultures isolated form Donor D to prepare a defined community, MET-1. Similarly, MET-2 was prepared from 33 species selected from Donor A. The breakdown of bacterial isolates found within MET-1 and MET-2 can be found in Table A1 and Table A2, respectively. Additionally, another 32 species from Donor D were selected to prepare the defined community, MET-2A. From the axenic fecal cultures isolated from Donor A, 32 species were selected to prepare the defined community, MET-3A. Once each of the bacterial strains were grown in pure culture using appropriate growth medium and conditions, an inoculum was created by combining biomass scraped from each culture plate and suspended in degassed 0.9% saline (37 °C). The final suspension volume for MET-1 was 30 mL while the final suspension volume for MET-2, MET-2A and MET-3A was 75 mL. The defined communities MET-2B and MET-3B were designed to be variants of MET-2A and MET-3A, respectively. MET-2B and MET-3B were prepared the same way as MET-2A and MET-3A, but differed from their respective parent cultures in that almost every isolate was added to the inocula in different proportions. The bacterial isolates for each of the four cultures and details of their biomass are detailed in Table A3. The same procedure for preparing defined community inoculants was repeated for the replicate experiment. Due to poor growth of four of the isolates in the pure cultures, the inocula for each of the defined communities had a different amount of bacteria for those poor-growth isolates as compared to the original experiment. These

differences are highlighted in Table A3. None of the defined community inocula included all four of the poor-growth isolates, so each defined community inocula only had 2-3 isolates that were different between the two replicate experiments. Given that the variant communities, MET-2B and MET-3B, had altered biomass in nearly every isolate, the deviations between the replicate experiments' inocula was considered to be negligible as any variations contributed by such differences would be indistinguishable from noise. A summary of the defined communities and the parent communities from which they were derived is shown in Table 3-3.

Table 3-3. Defined Communities and Their Parent Communities

Culture	Number of Bacterial species	Derived from Fecal Culture	Derived from Defined Culture
MET-1	33	Donor D	-
MET-2	33	Donor A	-
MET-2A	32	Donor D	-
MET-2B	32	-	MET-2A
MET-3A	32	Donor A	
MET-3B	32	-	MET-3A

Chapter 4 Metabolite Profiles of Media

4.1 Chapter Objective

While the process constructing metabolite profiles is relatively straightforward, doing so in an accurate and precise manner requires technical refinement. In this chapter, the techniques involved in obtaining the metabolite profiles of nutrient solutions is demonstrated, first through a step-by-step illustration, then in an experiment that exemplifies the application of the technique for the purpose of metabolomic analysis. The experiment presented in this chapter examined the metabolites of a semi-well defined media for culturing Chinese hamster ovary (CHO) cells and investigated the effects of UV irradiation of that media on the metabolite profile of the media and on the media's ability to support the culture of CHO cells. The analysis of semi-defined, unspent cell culture media is an appropriate platform to introduce the NMR-based metabolomic techniques developed thus far to analysis of nutrient solutions.

.

4.2 Constructing Metabolite Profiles

4.2.1 Section Objective

The purpose of this section is to illustrate NMR spectra of solutions containing a large number of metabolites and demonstrate how metabolite profiles are constructed from these convoluted spectra using Chenomx NMR suite software.

4.2.2 Processing NMR Spectra

The raw NMR spectra were processed manually as described in Chapter 3. Figure 4-1A depicts the raw NMR spectrum (0.9-1.5 ppm) of a defined media for CHO cells, where the signal peaks are out of phase, and neither baseline correction nor shim correction have been applied. Once the spectrum has been phase corrected (Figure 4-1B), baseline corrections can be made by setting the baseline manually. Figure 4-1C exemplifies the same NMR spectrum but with phase and baseline corrections applied. Figure 4-1D and Figure 4-1E show how shim correction is applied throughout the spectrum using the internal shape indicator, DSS. The blue DSS template is overlaid with and adjusted to fit the spectral DSS signal at 0.0 ppm. Mismatches between the spectral DSS signal and the software DSS template is considered to be a spectral artifact produced by imperfections in the NMR instrumentation. Therefore, all the NMR spectrum signals are altered to match DSS (Figure 4-1E). Figure 4-1F shows the NMR spectrum (0.9-1.5 ppm) after all the NMR processing steps. Each of these steps aid in producing the ideal NMR spectrum, without altering signal peak area which will facilitate signal identification and compound quantification in subsequent steps of building a metabolite profile.

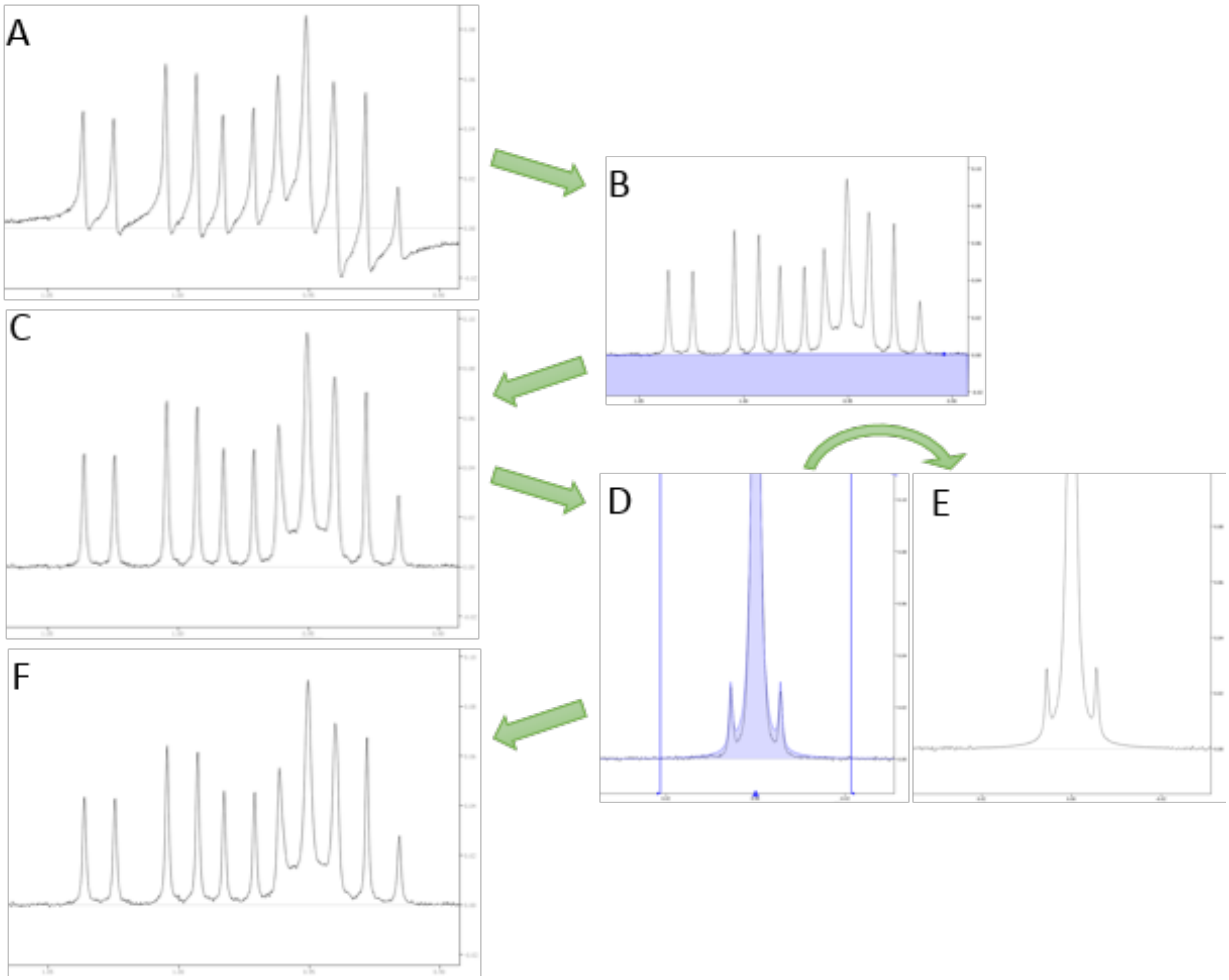

Figure 4-1. Effects of each spectral processing step on the NMR spectrum. The NMR spectrum within the range of 0.9-1.5 ppm of defined media for CHO cells is used as an example. (A) Raw NMR spectrum, no processing steps applied. (B) Applying phase corrections. (C) NMR spectrum after phase correction and baseline correction. (D) and (E) Applying shim correction to DSS signal at 0.0 ppm. (F) NMR spectrum after phase, baseline and shim corrections applied.

4.2.3 Building Metabolite Profiles

The assignment of compound identity to peak signals and quantifying those metabolite concentrations are performed using a targeted profiling technique. This method of interpreting the NMR spectra involves using the templates in the Chenomx NMR suite software library of compounds. By matching the compound templates to spectral signals, the identity of the compound is confirmed and the compound concentration is set (Figure 4-2). Assigning compound identities consistently for a given signal and setting compound concentration with high precision, even in areas of the spectrum with a high degree of signal convolution, are skills that are only developed with practice.

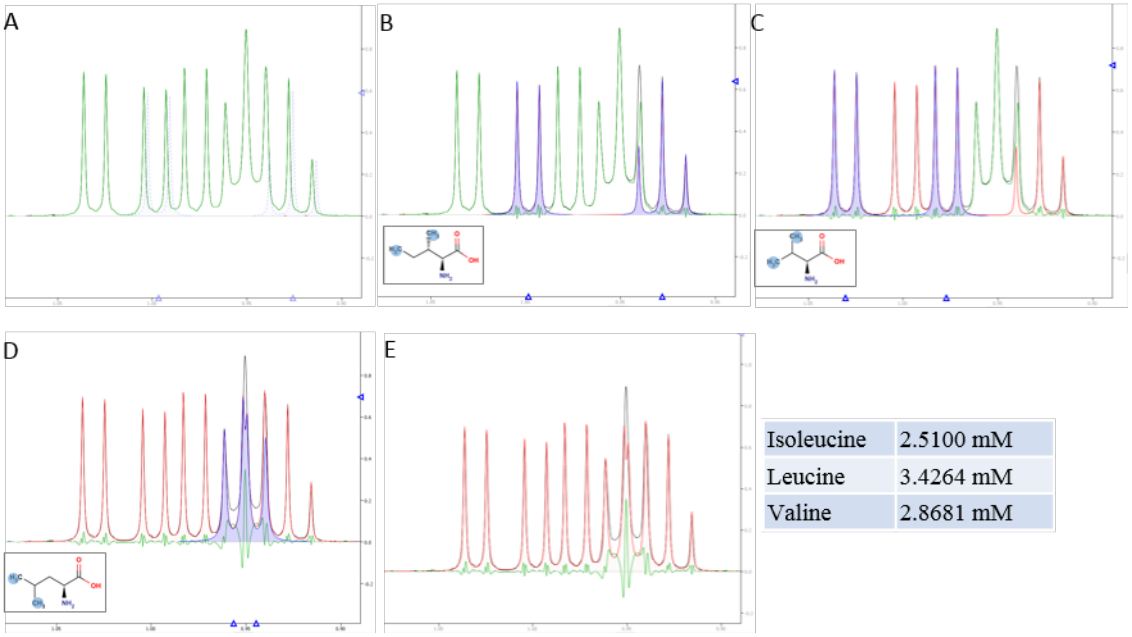

Figure 4-2. Identifying and quantifying compounds in NMR spectrum of defined CHO media (0.9-1.5 ppm). Black line represents the NMR spectrum. Blue dotted line represents the software's compound template, not yet fitted to the spectrum. Compound template fitted to the spectrum to confirm metabolite assignment and concentration to the corresponding signal is shown in transparent blue. Red line represents the sum line, showing the area of the spectrum that has already been fit with a compound, where overlaying templates have an additive effect. Green line represents the subtraction line, which is the difference between the NMR spectrum (black) and the profiled compounds (red). (A) Non-profiled spectrum. (B) Isoleucine. (C) Valine. (D) Leucine. Only clusters between 0.9 and 1.5 for each compound are shown. Signal clusters outside this range would also be used to determine compound identity and concentration.

4.3 Treating Cell Culture Media with UV Irradiation against Adventitious Agents: Minimal Impact on CHO Performance

4.3.1 Journal Article Authorship

Sandi Yen[1,#], Stanislav Sokolenko[1,#], Bhavik Manocha[1], Ankit Patras[2] , Farnaz Daynouri-Pancino[2], Eric JM Blondeel[1], Michael Sasges[2], Marc G Aucoin[1,*]

[1]Waterloo Institute for Nanotechnology, Department of Chemical Engineering, University of Waterloo,

Waterloo, Ontario, Canada, N2L3G1

[2]Aquafine Corporation, Valencia CA, USA, 91355-4198

#: contributed equally to this work

***corresponding author:** Marc G. Aucoin, Department of Chemical Engineering, University of Waterloo, Waterloo, ON, CANADA N2L 3G1, Tel: 1-519-888-4567 x36084, Fax: 1-519-888-4347, e-mail address: maucoin@uwaterloo.ca

4.3.2 Justification of Original Work

The following journal article was published in Biotechnology Progress on July 29, 2014. The experiment was a collaborative project between the Aucoin group at the University of Waterloo and the Sasges team at Trojan Technologies (London, ON, Canada)/Aquafine Corporation (Valencia, CA, USA). UV irradiation of solutions and related protocols were conducted by Trojan UV, while metabolite analyses and cell culture experiments were performed by the Aucoin group of University of Waterloo. The objective of this experiment was to investigate the effect of UV irradiation on the chemical composition of cell culture media and to determine the level of irradiation that can be

tolerated before negative effects are observed in simple batch growth cell culture. Within the Aucoin group the contributions towards this project must be credited to several team members under the supervision of Marc Aucoin. The cell culture responsibilities were shared between Sandi Yen and Bhavik Manocha. Antibody analyses were performed by Jann Catherine Ang and metabolite profiles of cell culture media were constructed by Sandi Yen. Data analysis and interpretation was performed by Stanislav Sokolenko. The overall manuscript was written primarily by Sandi Yen and Stanislav Sokolenko, with editing provided by Eric Blondeel. This work is included in this thesis because establishing metabolite profiles of unspent defined media was the initial step to being able to build metabolite profiles of complex solutions to be encountered throughout the body of the thesis. Unspent defined media established foundational skills in NMR-based metabolite analysis because it is a solution that contains many metabolites, thus producing complex NMR spectra. At the same time, the known composition of the media provided information on the accuracy and sensitivity of the metabolite profiles. In this respect, the experiment presented here fits appropriately as the preliminary work within this body of research. With minor editorial changes to fulfill formatting requirements, this chapter is substantially as it has been submitted and published in *Biotechnology Progress.*

4.3.3 Abstract

Sterility of cell culture media is an important concern in biotherapeutic processing. In large scale biotherapeutic production, a unit contamination of cell culture media can have costly effects. Ultraviolet (UV) irradiation is a sterilization method effective against bacteria and viruses while being non-thermal and non-adulterating in its mechanism of action. This makes UV irradiation attractive for use in sterilization of cell culture media. The objective of this study was to evaluate the effect of UV irradiation of cell culture media in terms of chemical composition and the ability to grow cell cultures in the treated media. The results showed that UV irradiation of commercial cell culture media at

relevant disinfection doses impacted the chemical composition of the media with respect to several carboxylic acids, and to a minimal extent, amino acids. The cumulative effect of these changes, however, did not negatively influence the ability to culture Chinese Hamster Ovary cells, as evaluated by cell viability, growth rate and protein titre measurements in simple batch growth compared to the cells cultured in control media exposed to visible light.

4.3.4 Introduction

Sterility is an ongoing challenge for commercial bioprocesses, particularly in the production of biotherapeutics, where sterility assurance levels are mandated across entire manufacturing processes. Major pharmaceutical companies have suffered significant losses due to both bacterial and viral contaminations at various stages of production[93–95]. Such incidents call attention to the need to address the risk profile associated with biopharmaceutical products. As an ingredient in the production process, culture media is a potential source of contamination that can propagate downstream and ultimately affect patient safety. Complementary media sterilization methods can confer additional safeguards for biopharmaceutical production and act as a viral barrier to mitigate contamination risks.

Sterility assurance levels of 6, or a probability of 10^{-6} that a single viable microorganism survives the sterilization process and is present in the final product, is the level of sterility that has become the standard requirement for pharmaceutical products and sterilization processes[96]. This level of sterility is traditionally accomplished via terminal sterilization processes by high heat and pressure[97,98], which can be inappropriate for solutions having heat-labile components. High-temperature/short-time (HTST) pasteurization is also widely used as an upstream viral barrier. While this technology has been shown to be effective in inactivating viruses in media[99,100], HTST has been shown to be incompatible with certain serum containing media[100]. Applying HTST can also cause

phosphate- and calcium-based precipitates to form, ultimately impacting the operation of the HTST system[101,102].

As an alternative to heat treatments, aseptic procedures require all production equipment and components to be either ‘single-use’ disposable or steam-sterilized before use. Cell media are often subjected to sterile filtration with 0.22 µm filters following aseptic steps to further preclude media contamination[97].

Ultraviolet (UV) radiation is known to be an effective sterilant against various biological contaminants, including bacteria and viruses[103–106]. For example, active cytomegalovirus (CMV), can be reduced by one log with a UV fluence of 5 mJ/cm^2 [107], while minute virus of mice (MVM) requires only 2 mJ/cm^2 [108,109]. More resistant viruses, such as adenovirus require up to 306 mJ/cm^2 to achieve a log inactivation of approximately 6 [15,16].

UV disinfection is favourable for its non-thermal and non-adulterating characteristics, and has been adopted by several industries, including the biopharmaceutical industry, for packaging and surface sterilization applications[112–115]. UV irradiation is able to approach the sterilization criteria of 6-log reduction with MS2 bacteriophage, which is used as a model challenge organism and used to asses reduction equivalent fluence of a UV system[116–118]. This makes UV irradiation a sterilization process fitting for diverse biopharmaceutical applications, such as in the sterilization of polyethylene bottles as well as the disinfection of air, water and surfaces[104].

The application of UV irradiation to cell culture media as a sterilization method has been limited thus far and conducted primarily in industry. Previous work on UV irradiation of media has suggested that treatment can result in changes to substrate concentrations or the production of various by-products[119], possibly leading to cell growth failure[120] or changes in critical quality attributes

(CQA) of the final product. The objective of this work was to investigate the effect of UV irradiation on the chemical composition of cell culture media and to determine the level of irradiation that can be tolerated before negative effects are observed in simple batch growth cell culture.

4.3.5 Materials and Methods

4.3.5.1 Experimental design

Two culture media were evaluated in this study: a protein-free, serum-free, chemically defined medium optimized for the growth of Chinese hamster ovary (CHO) cells and expression of recombinant proteins in suspension culture, as well as Media 199 (M199), formulation 11150, a fully defined protein-free, serum-free media used to culture chick embryo fibroblasts. Four UV irradiation doses were considered in this work, 110 mJ/cm^2, 180 mJ/cm^2, 250 mJ/cm^2, and 400 mJ/cm^2. Due to the wide range of UV fluence at which bacteria and viruses are inactivated, these high level dosages were selected to be effective against both bacteria, which are more UV-sensitive, and viruses, which tend to be more UV-resistant[110]. Fluence was estimated based on the log-reduction of MS2 bacteriophage in the media given that 110 mJ/cm^2 yields a 5-log reduction of this organism.

To best characterize the overall chemical changes in media, we used a nuclear magnetic resonance (NMR)-based metabolomics approach to characterize a large variety of media metabolites that could potentially be affected by the UV treatment. To achieve a high level of confidence in the observed effect of UV treatment on metabolite concentrations, each treatment was repeated 3-6 times per sample set, with multiple sample sets generated over a 4 month period. This allowed the separation of observation and process variability from the quantification of treatment effect.

The impact of UV irradiation on cell growth was determined by culturing cells in treated media. Two controls were used for the analysis: unmodified media that had not been treated in any

way, which is referred to as samples receiving 0 mJ/cm^2, and non-irradiated samples that had been stirred and exposed to air for the same duration as the UV-irradiated samples.

4.3.5.2 Media irradiation

Irradiation of media was executed by Trojan Technologies (London, Ontario, Canada). A collimated beam apparatus equipped with a low-pressure mercury lamp emitting at 254 nm was used for uniform and quantified UV irradiation of media[121]. The calibration, fluence determination and quality assurance protocols associated with UV irradiating solutions were carried out according to standardized practices in the water disinfection industry[122]. The media was irradiate at low volumes, ~5 mL, in 10-mL beakers with stirring to ensure uniform UV delivery throughout the liquid solution. The actual UV fluence delivered to the media was calculated based on the measured UV dose emitted by the collimated beam and the optical properties of the media being treated.

4.3.5.3 Verification of UV fluence

In order to establish the actual UV fluence values delivered to the culture media, a viral clearance test was conducted using a challenge organism inoculated into the culture media. The challenge organism, MS2, is a single-stranded RNA virus, with an icosahedral shell approximately 27 nm in diameter, used extensively in validation of UV disinfection systems for drinking water[123].

MS2 bacteriophage was suspended in phosphate-buffered saline and irradiated in a collimated beam to various UV fluence values. The buffer used in this characterization had high transparency at 254 nm, typically 90% at 1 cm, so that the UV intensity gradient in the fluid was small, and therefore the uncertainty in the UV fluence was small. The 254 nm irradiance at the sample surface was measured using a radiometer with NIST-traceable calibration. The UV fluence gradient caused by optical absorbance of the fluid, along with the effects of beam divergence, non-uniformity, and

surface reflection were all accounted for as recommended in the standard method. The samples were stirred during irradiation to ensure that all organisms were exposed to the same integrated UV fluence over the course of the irradiation. After irradiation, each sample was diluted serially and aliquoted into culture tubes containing 1 mL *Escherichia coli* broth culture and 20 mL of molten tryptone yeast extract glucose agar containing triphenyl tetrazolium chloride. The mixtures were mixed by inversion and plated into sterile Petri dishes. The agar was allowed to solidify and the plates were incubated at 35_C60.5_C for 18–24 h before performing a plaque assay. The plates were then evaluated to determine the number of plaque forming units. By comparing with the control (non-irradiated) sample, the relationship between UV dose and the log-reduction of this population of MS2 was established.

With the UV-sensitivity of the organism determined, the UV dose applied to the culture media was determined by using MS2 as the dose indicator. This is sometimes denoted as "Reduction Equivalent Dose (Fluence)," since it is inferred from the log reduction of a well-characterized challenge organism. MS2 from the characterized population was inoculated into samples of the culture media, and the optical properties of the inoculated media were measured. The optical properties of the media were used to calculate the irradiation times necessary to achieve a desired UV dose in the collimated beam apparatus. The samples were then irradiated for the prescribed time, and then serially diluted and cultured as described above. The log reduction in numbers of active MS2 were used to calculate the UV dose received by the media in each irradiation by using the sensitivity of the MS2 as established by the buffer tests. The resulting Reduction Equivalent Dose vs. target dose has been plotted and may be seen in Figure 4-3. This relationship was used to determine the UV fluence values used in subsequent irradiations of mammalian cell culture media.

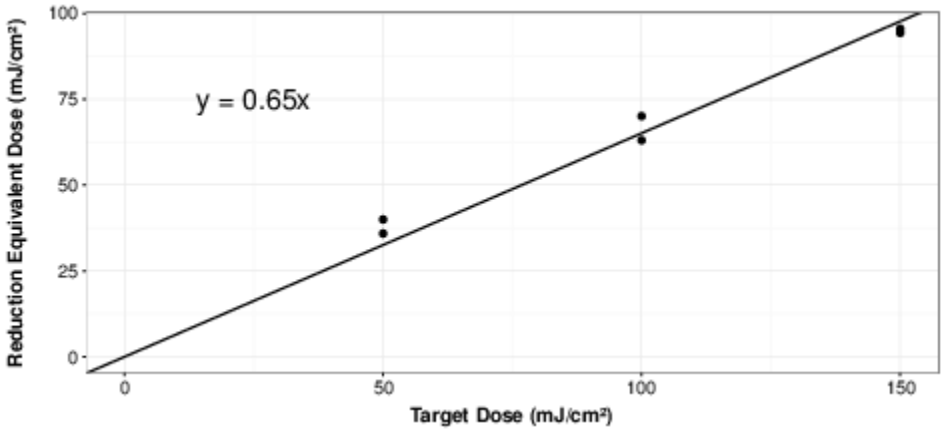

Figure 4-3. Received UV fluence in commercial media measured by bioassay (MS2 bacteriophage) vs. target value. The "Reduction Equivalent Dose (R.E.D.)" applied to the culture media was determined by well-characterized MS2 phage as the dose indicator (Target dose).

4.3.5.4 Cell culture

CHO^{BRI} cells, obtained from the National Research Council Canada, were suspension-adapted cells grown in proprietary, serum-free growth media blend, BioGro-CHO, supplied by BioGro Technologies Inc. (Manitoba, Canada). The cells were adapted to grow in a commercial serum-free, chemically-defined CHO media, supplemented with GlutaMAX™ (Invitrogen Corp., Burlington, Canada) and HT Supplement (Invitrogen Corp., Burlington, Canada). The parental culture was routinely maintained in shaker flasks kept in a humid incubator (37º C, 5% CO_2) and agitated at 100 rpm. Both the parental and experimental cultures were cultured in 125 mL, graduated non-pyrogenic polycarbonate Erlenmeyer flasks (Corning Inc., NY, USA). Once the mother flask reached a viable cell density of 2-3x10^6 cells/mL, the experimental cultures were inoculated at a seeding density of 0.2x10^6 viable cells/mL.

For culture tests, the irradiated and control samples of CHO media were supplemented with GlutaMAX™ and HT Supplement after media irradiation and prior to inoculation. GlutaMAX™ and HT Supplement were not subjected to UV irradiation in these experiments. Multiple irradiated samples were pooled to achieve 15-20 mL volumes required for culture growth and supplemented as above.

4.3.5.5 Cell count

Each experimental culture was counted at 24-hour intervals for five days using both the classic haemocytometer counting method, and with a Coulter Counter Z2 (Beckman-Coulter, Miami, USA) for cell size distribution analysis. Samples counted in the haemocytometer were prepared in a 1:2 dilution with Trypan blue (10% w/v in PBS) before loading the diluted sample into the counting chamber. In circumstances where the cell count was greater than 3-4 x 10^6 viable cells/mL, the sample was diluted by a factor of 10 using CHO media as the diluent, while maintaining a 50:50 ratio

of dye to sample and diluent. Cell size distributions were obtained from a Coulter Counter Z2 by loading samples diluted 100-fold in Isoton® II solution. Cell viability was also calculated daily based on haemocytometer counts.

4.3.5.6 NMR spectroscopy and metabolite profiling

NMR samples were prepared by the combination of 630 μL of media and 70 μL of internal standard composed of 99.9% D_2O with 5 mM 4,4-dimethyl-4-silapentane-1-sulfonic acid (DSS) serving as a chemical shape indicator (CSI) and 0.2% w/v sodium azide to inhibit bacterial growth (Chenomx Inc., Edmonton, Canada). The solution was vortexed and pipetted into 5 mm NMR tubes (NE-UL5-7, New Era Enterprises Inc., Vineland, NJ, USA) for scanning. NMR spectra were obtained with a 600 MHz Bruker Avance spectrometer, equipped with a Triple Resonance Probe (TXI 600). The spectra were acquired using the first increment of the NOESY pulse sequence with a 1-second presaturation pulse, followed by a 4-second acquisition time. All spectra processing was carried out with Chenomx NMR Suite 7.5 (Chenomx Inc., Edmonton, Canada). Baseline, phase, shim, and chemical shape corrections were all performed manually using tools available with the software. Briefly, baseline correction was carried out by the selection of cubic spline points to subtract from the observed spectra, while shim correction corresponded to reference deconvolution to remove line asymmetry[124]. Subsequently, compounds were quantified by targeted profiling. The observed spectra were superimposed with Chenomx's built-in library of chemical resonances, with metabolite concentration quantified using DSS as a reference compound (for more information on targeted profiling, see Weljie et al.[31]).

4.3.5.7 Antibody titer

After 5 days of culturing, the supernatant for the CHO cultures were collected. Each acquired sample was concentrated down to 1 mL volume using a Centriprep Centrifugal filter unit with an Ultracel-10

membrane (EMD Millipore, MA, USA). The concentrated samples were passed through a Protein A HP Spin Trap column (GE Healthcare) and purified IgG antibodies were collected. Antibody titer (total extracellular protein) was determined using a Pierce™ BCA protein assay kit (Pierce Biotechnology, Rockford, USA).

4.3.6 Results

4.3.6.1 NMR metabolite analysis

Twenty-eight compounds were identified and quantified in the media by NMR, of which twelve compounds were found to have statistically significant trends with respect to UV irradiation: pyruvate, acetate, arginine, lysine, threonine, tryptophan, tyrosine, choline, methionine, formate, ethanolamine and pyridoxine (in decreasing order of concentration change). When the concentration profiles of these compounds were compared to changes from exposure to visible light and air (without UV) in the control samples, it was found that five compounds had similar changes from visible light and air exposure: arginine, threonine, choline, methionine, and ethanolamine. These changes may be related to exposure to oxygen or to the effects of room light, but are not related to UV exposure, since there was negligible UV exposure in these controls. Concentrations of pyruvate, acetate, and formate were also observed to change with exposure to visible light and air, albeit much less than under UV treatment. Therefore, the observed concentration changes of only seven compounds could be directly linked to UV exposure - pyruvate, acetate, lysine, tryptophan, tyrosine, formate, and pyridoxine. The concentration profiles of these compounds were all approximately linear as a function of UV fluence (Figure 4-4). The specific concentration changes can be seen in Table 4-1, calculated per 100 mJ/cm^2 of UV fluence. Amino acids changed the least in relative concentration, ranging from decreases of 3% to 7%. A much greater effect was observed for the carboxylic acids, with acetate concentration, for example, increasing twofold.

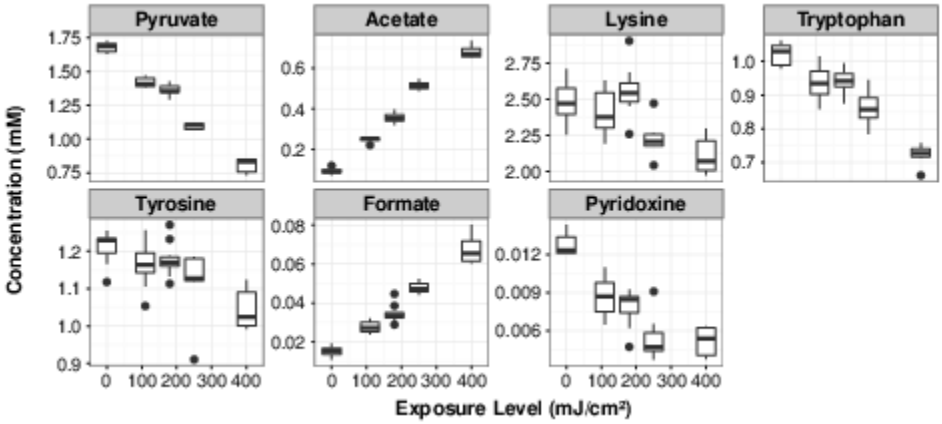

Figure 4-4. Compound concentration as a function of UV fluence. The box-plots contain aggregated data from multiple experiments in which exposures were performed 3-6 times at each level.

Table 4-1. Changes in compound concentration per 100 mJ/cm^2 of UV dose.[2]

Compound	Concentration Change (μM)	Concentration Change (%)
Pyruvate	-218.7 +/- 6.6	-13.0 +/- 0.4
Acetate	152.1 +/- 3.4	164.0 +/- 3.6
Lysine	-91.5 +/- 21.4	-3.7 +/- 0.9
Tryptophan	-70.6 +/- 5.6	-6.9 +/- 0.5
Tyrosine	-39.6 +/- 7.2	-3.3 +/- 0.6
Formate	13.2 +/- 0.5	86.5 +/- 3.5
Pyridoxine	-2.0 +/- 0.2	-15.6 +/- 1.6

[2]Linear regression was performed on the change in compound concentration as a function of UV dose with the presented number corresponding to the predicted concentration change per 100 mJ/cm^2 (with the +/- representing standard deviation).

4.3.6.2 Cell culture

Cell culture performance was primarily assessed by viability, growth rate, and final cell concentration. Controls (exposed to visible light) were cultured alongside cultures grown in UV-treated media to account for changes in cell behaviour. Overall, UV-treatment of the media did not have a significant effect on cell growth. Therefore, any variability observed in cell growth and protein production can be attributed to random effects imparted by the experimental environment, such as visible light. Cell viabilities of both treated and control samples were routinely above 95%, with no significant difference between the two groups as tested by paired Student's t-test at the 95% confidence level. Additionally, no differences were observed in cell size distributions (data not shown). Growth rates can be seen in Figure 4-4A and the final cell concentrations in Figure 4-5B. Overall, the differences between treated and control cultures were not statistically significant.

In the course of this study, quail muscle fibrosarcoma (QM5) cells were also cultured in M199 media treated with a dose of 110 mJ/cm^2. No differences were observed between QM5 cells grown in UV-treated and those grown in control media. In both cases, the cells were able to reach confluence within 4 days, with no apparent changes in morphology. Higher UV doses were not assessed for this media.

4.3.6.3 Protein titre

Similar to the growth data, recombinant IgG protein titers from the CHO cells were found to be very similar for UV-treated and control media. Comparing the yield on a purely volumetric basis suggested that UV-treated media may result in somewhat better protein production (Figure 4-5C). However, when integrated viable cell density was used to calculate specific protein production levels on a per-cell basis[125], there were no significant differences in protein yield (Figure 4-5D).

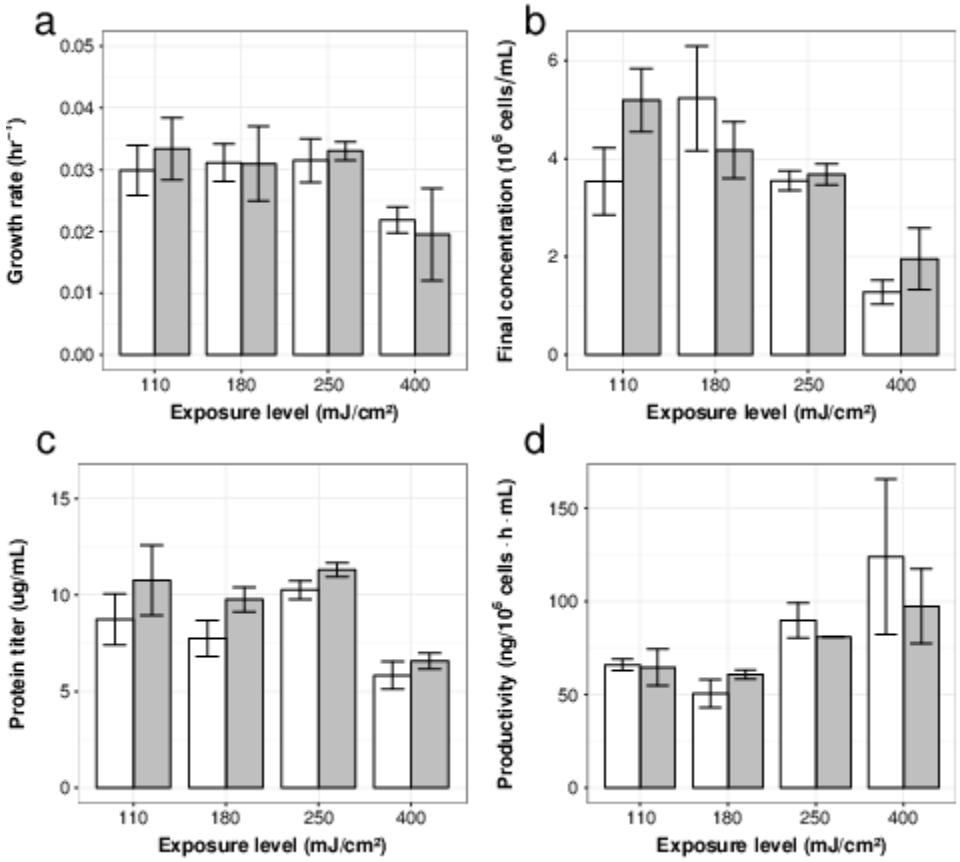

Figure 4-5. (A) Comparison of cell culture performance on control (white) and treated (grey) media. In all cases, controls (exposed to visible light) were cultured alongside cultures grown in UV-treated media to account for changes in cell behaviour over a 4-month period. No statistically significant differences were observed between treated and control media for any of the culture parameters, as assessed with a two-sample t-test at a 95% confidence level. a) Mean exponential phase growth rates (with standard errors) for cell cultures grown in UV-treated and control media. Values were calculated from the log-regression slope of concentration data from days 1-3. (B) Mean final cell concentrations (with standard errors) for cell cultures grown in UV-treated and control media. For the 250 mJ/cm^2 dose, the cultures grown on treated media resulted in nearly identical final concentrations, leading to overlapping error bars. (C) Mean protein yield (with standard errors) for cell cultures grown in UV-treated and control media. (D) Mean specific protein yield (with standard errors) for cell cultures grown in UV-treated and control media. Values were calculated by dividing the volumetric protein yield by the integrated viable cell density. For the 250 mJ/cm^2 dose, the cultures grown on treated media resulted in nearly identical specific protein yield, leading to overlapping error bars.

4.3.7 Discussion

Although UV irradiation has been previously reported to negatively impact cell growth[120], the limiting UV dose values were not indicated. No negative effect was observed in this work for either CHO cells grown in UV-irradiated CHO medium or QM5 cells grown in UV-irradiated M199, suggesting that the impact of UV-irradiation cannot be generalized, or the effect is only encountered when pushing the culture beyond a simple single batch process. The metabolic changes observed due to UV irradiation of media did not impact the overall pH measurement of the media solutions used. It should be noted that we have observed pH shifts in other irradiated media (data not shown), though none in this work. This is in part due to the sodium bicarbonate buffer in the media and the relatively small absolute changes of impacted compounds, such as acetate.

Observed changes in substrate concentrations were also limited. Amino acid concentration changes that did occur were on the order of 1-7% per 100 mJ/cm^2. Most cell culture media contain these compounds in excess so the extent of degradation of amino acids from UV irradiation, even at high UV doses, is unlikely to impact culture performance. The small changes in substrate concentration also meant that any specific degradation products were generally masked by overall compound variability, which we have previously found to be approximately 5%-10% for most compounds, measured as a coefficient of variation[39] . Pyridoxine was the only vitamin that could be observed with NMR, and therefore no general conclusions can be made about overall vitamin stability. However, the cell growth data indicates that the level of vitamin degradation was too low to impact cell growth compared to our control.

The largest changes in compound concentration in these media were observed for carboxylic acids. Pyruvate concentrations were found to decrease by approximately 13% per 100 mJ/cm^2, with acetate and formate increasing by 160% and 90% per 100 mJ/cm^2, respectively. This corresponded to

absolute changes of 218 μM, 152 μM, and 13 μM for pyruvate, acetate, and formate respectively. The three compound concentrations showed parallel trends in both control (visible-light-exposed) and treated media. Considering the high degree of precision in its quantification (2% coefficient of variance including set-to-set variability), pyruvate concentration can be seen as an attractive marker for assessing the overall effect of UV treatment in cell culture media using NMR spectroscopy. As previously mentioned, no additional impact on cell growth was observed compared to the control in spite of such changes to these metabolites.

This decrease in pyruvate is not altogether surprising. Pyruvate has been previously identified to be a prominent antioxidant in biological systems and has been found to yield acetate when reacting with peroxynitrite, a powerful oxidant[126], or hydrogen peroxide[127]. It may serve a similar role in the quenching of oxidative products formed during UV treatment. Based on absolute concentration changes, it would appear that acetate is the major product of such reactions, although formate may also be involved.

The UV doses applied in this study would achieve disinfection for most adventitious agents relevant to CHO cell culture, such as mouse minute virus and reovirus[95,103,110]. Since filtration is seen as an effective method to remove larger virus and bacteria, UV disinfection is primarily seen as a complementary method to inactivate adventitious agents not captured by filtration.

4.3.8 Conclusion

The UV doses applied in this study did not functionally impair the culture media tested, as measured by cell growth and antibody production of CHO cells. Cell morphology was not impacted and the cell size distribution was comparable between cultures grown in UV-treated media and control media that had only been exposed to visible light. These findings were consistent at various fluence levels, even

at very high exposure levels of 400 mJ/cm^2. QM5 cells were also unaffected by media irradiation, though media was only exposed to UV doses up to 110 mJ/cm^2. Changes to major media components were largely limited to carboxylic acids, with only minimal changes in amino acid concentrations. The findings in this study serve as a demonstration of some of the changes that can occur to cell culture media after applying UV sterilization. While these results cannot be generalized to all cell lines and culture media, this study provides insight into the effects of UV-sterilization in cell culture applications. Further validation with investigations focusing on longer culture duration, higher cell densities, repeated passaging in UV-treated media, and ultimately the effect on protein product critical quality attributes is currently being undertaken.

4.3.9 Acknowledgements

This work was supported in part by NSERC ENGAGE and NSERC Strategic Network (MabNet) Grants to MGA and NSERC Canada Graduate Scholarship to SS. The authors also thank Jann Catherine Ang for quantifying the antibodies produced in this study.

Chapter 5 Standardization of Fecal Water Samples and Analysis

5.1 Chapter Objective

The focus of this chapter is on setting a level of standardization on reporting on the fecal metabolome. Fecal water is the aqueous phase of fecal samples that is prepared by adding water (buffer) to feces, mixing and recovering the liquid phase after removing the solids and centrifugation. Even though fecal water is the accepted means of studying gut microbiota[2–4], there is still a need to developing a standard means of preparing and reporting on the biofluid for metabolomic purposes. Granted, there have been investigations into various approaches for studying fecal water using other metabolomic tools[20,21]. To develop an appropriate standard for reporting on fecal metabolome, metabolite profiles of fecal water obtained from varying water to feces ratio were examined to determine the minimal concentration of feces required for metabolite analysis of fecal water, and propose normalization techniques for reporting on fecal metabolomes.

5.1.1 Materials and Methods

5.1.1.1 Preparation of Fecal Water Samples

Fecal water samples were prepared as follows. Stool samples were collected from Donor E, a neurotypical, physiologically healthy female (Table 3-3) with no dietary restrictions or any use of antibiotics within 6 months of sample collection. After collection, samples were stored at -80 °C. Frozen stool samples were thawed and the desired amount was weighed into a bag, along with autoclaved PBS. The stool-PBS mixture was homogenized in a stomacher for 1 minute, and the slurry was subsequently ultracentrifuged for 2 hours at 50 000 rpm, 4°C. The supernatant was extracted by needle and syringe. The supernatant in the syringe was shaken vigorously and then passed through 0.22 µm syringe filter. The samples were prepared in increasing concentrations of feces, between the ranges of 8-35% (w/v) (Table 5-1), and aliquoted into duplicate samples. Fecal water samples were stored at 4°C until transport (on ice) to the Aucoin lab the next day. The samples were stored at -80 °C upon receipt of the samples until NMR analysis 12 days later. On the day of NMR scanning, fecal water samples were thawed at room temperature and prepared for NMR spectroscopy as outlined in the Chapter 3.

Table 5-1. Fecal concentration of fecal water samples. Duplicate samples were provided for each fecal concentration.

Mass of raw stool samples (g)	Volume of PBS (mL)	Fecal concentration (g/mL)	Fecal concentration (w/v %)
2.6	30	0.09	9%
5.6	30	0.19	19%
15.2	60	0.25	25%
8.2	30	0.27	27%
9.1	30	0.30	30%
20.6	60	0.34	34%

5.1.1.2 Statistical Analysis

Trends in metabolite concentration due to changes in fecal concentration were evaluated for linearity by using linear regression. Correlations were evaluated with Pearson product-moment correlation coefficient, *r*. Normalizing compound concentrations to be independent of fecal concentration was calculated by dividing metabolite concentration by fecal concentration. Normalizing compound concentrations to acetate, an internal metabolite was calculated by dividing the metabolite concentrations by the concentration of acetate from the same metabolite profile.

5.1.2 Results

5.1.2.1 Fecal Water Analysis and Sample Preparation

In total, 39 compounds were identified and quantified in each of the metabolite profiles of the fecal water samples. An overview of the profiles indicated that there was a linear relationship ($p \leq 0.05$ at the 95% confidence level) between the fecal concentration and the metabolite concentrations. Furthermore, there was a strong positive correlation (correlation coefficient > 0.74) between the fecal concentration at which the samples were prepared and the observed compound concentrations (Figure 5-1). The only compound that did not follow this trend was ethanol. Ethanol did not have a significant linear relationship between compound concentration and fecal concentration. The changes in compound concentrations were apparent even by comparing the NMR spectra directly (Figure A1).

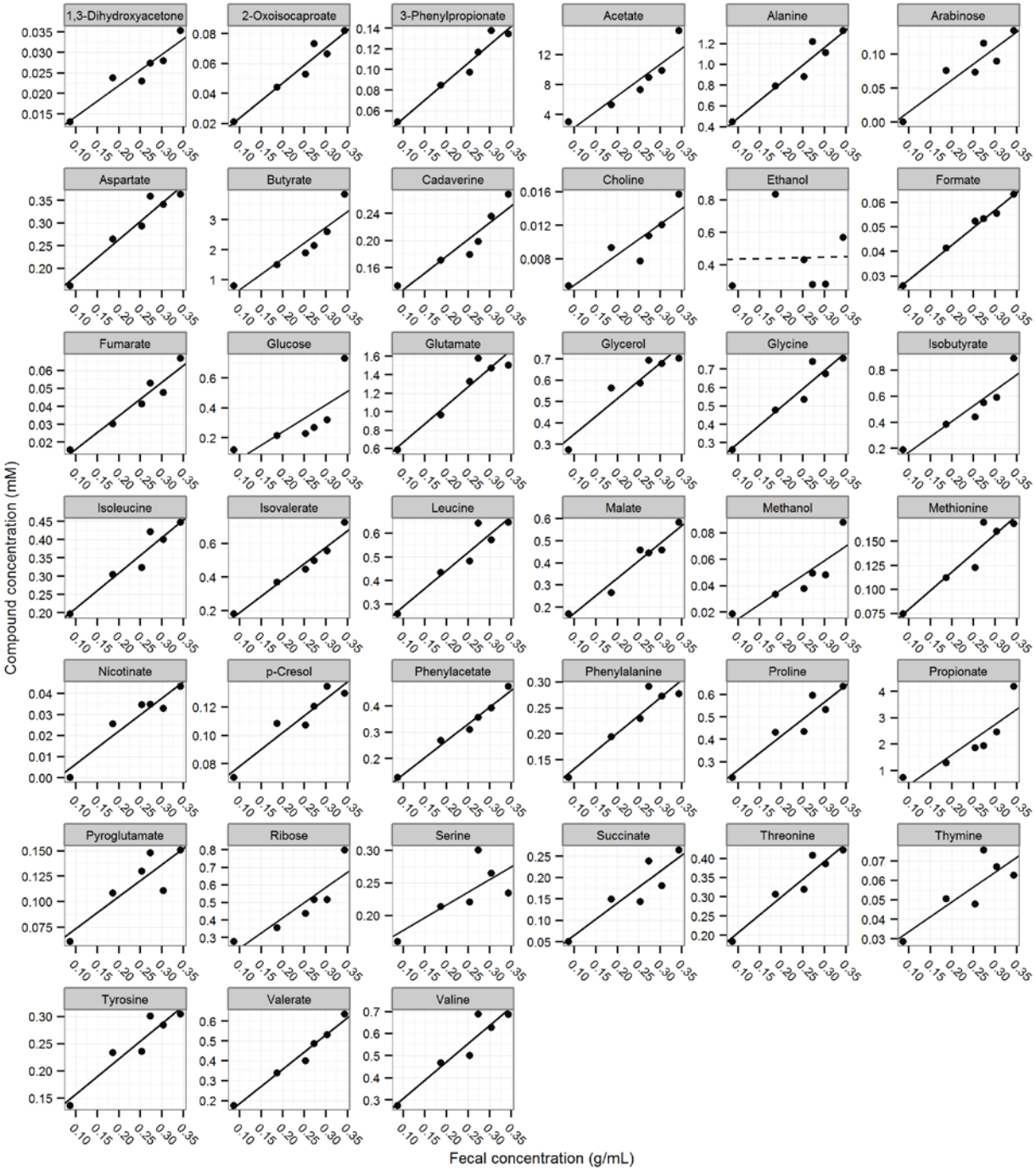

Figure 5-1. Metabolite profile overview. Compounds exhibiting significant linear relationship and a strong correlation between metabolite concentration and fecal concentration are identified by illustrating the trends in a solid black line. Otherwise, a dashed line is used. Mean concentration from duplicate samples shown.

5.1.2.2 Normalization of Metabolite Concentrations

The linear relationship between metabolite concentration and fecal concentration means that metabolite concentrations can be normalized as metabolite concentrations per unit of feces. Dividing the metabolite concentration by fecal concentration expresses the metabolites in terms of mmol/g of feces. Given that all the samples examined here were identical in composition and different only in their dilution, the normalized metabolite concentration should be very similar across different fecal concentrations. Indeed, the spread of the normalized values is quite narrow for each compound (Figure 5-2A).

Alternatively, metabolite concentrations can be normalized to an internal compound such as acetate. Since acetate was a compound that was consistently found at high levels (~4 times larger than the next most abundant compounds), it is an attractive marker to gage the contents of the fecal samples. Normalizing the metabolite concentrations to acetate was performed by dividing each metabolite concentration by the mean acetate value for each fecal concentration. Reporting compounds as a percent of acetate presented the metabolite concentrations independent of fecal concentration. The success of acetate-normalization was demonstrated in the narrow range observed in the acetate-normalized values for every compound. (Figure 5-2B).

In comparing the efficacy of each type of normalization, it was noticeable that the spread of normalized values, as measured by range, in acetate-normalized values is generally smaller than that of the normalization by fecal concentration. The reason that range is used for the comparison instead of standard deviation was because standard deviation is a measurement that is magnitude dependent. Inherent to the normalization method, acetate-normalized values must be between 0 and 1, while the fecal concentration-normalized values must be within 1-3 orders of magnitude of the original mM

concentration. The scale differences between the two types of normalization limit the meaningfulness of comparing the standard deviation values directly.

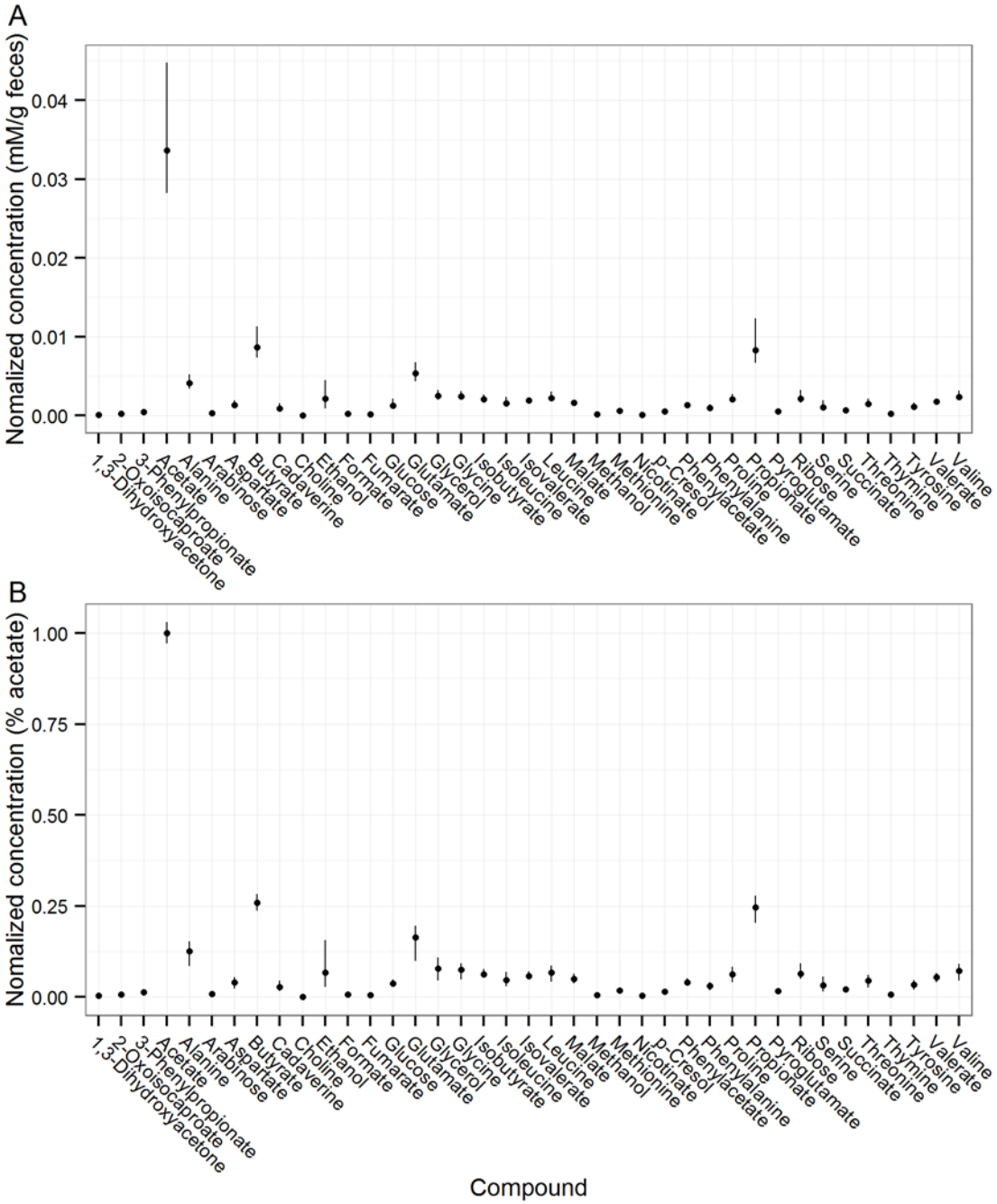

Figure 5-2. Normalized compound concentrations. Bars represent range of normalized values. (A) Normalized by fecal concentration. Calculated by dividing observed compound concentration (mM) by fecal concentration (g/mL). (B) Normalized by mean acetate concentration. Calculated by dividing observed compound concentration (mM) by mean acetate concentration for each fecal concentration.

5.1.2.3 Impact of NMR Spectra on Metabolite Profiles

As mentioned before, 39 compounds were included in the metabolite profiles of the fecal water samples. Even though all 39 compounds were assigned and quantified with relatively high confidence, certain compounds became increasingly difficult to profile as fecal concentration decreased. The decreased resolution, related to the signal-to-noise ratio of the spectra, can yield larger variance in compound quantification, or even impair proper assignment of compound identification. Examples of compounds that had signals that were sensitive to low fecal concentrations included proline, choline, pyroglutamate, p-cresol, arabinose, and glucose (Figure 5-3, Figure A2). For example, proline and glucose both had clusters within the 3.2-3.55 ppm range of the spectrum that were crucial for both identification and quantification of these two compounds, as they are relatively free of convolution with other metabolite signals in this region. As seen in Figure 5-3, proline and glucose clusters were far less prominent relative to the spectrum's noise levels in the low fecal concentration sample compared to the high fecal concentration sample. So, in the spectra representing higher concentrations of feces, compounds such as proline and glucose could be assigned and quantified with relatively high accuracy and confidence. However, if the low fecal concentration samples were profiled independently, i.e. in the absence of also profiling high fecal concentration samples, these compounds could be lost due to the lack of evidence for profiling those subtle signals.

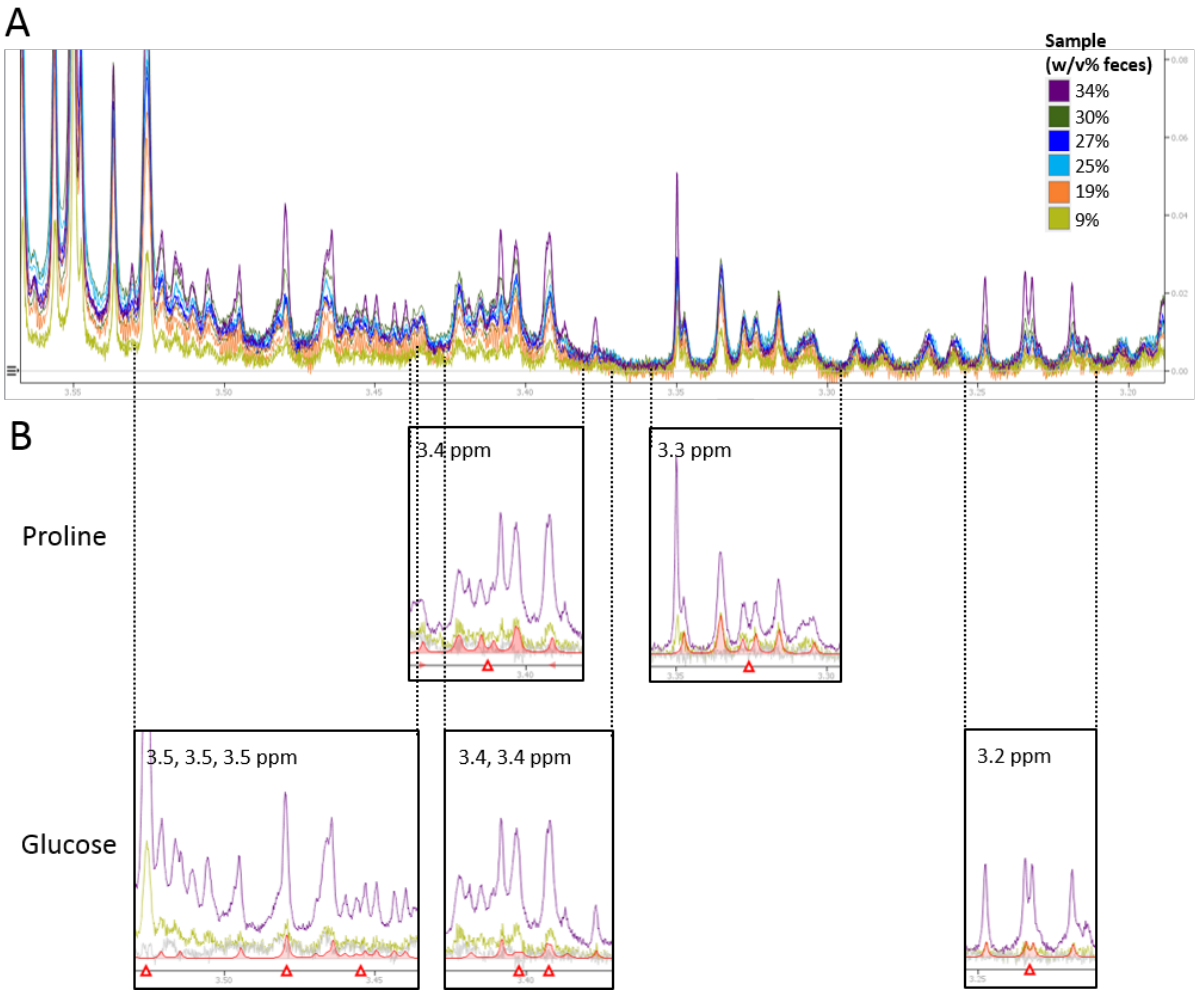

Figure 5-3. Profiling confidence impacted for certain compounds. (A) Spectrum overlay of metabolite profiles of samples for different fecal concentration (3.2-3.55 ppm). (B) Comparison of spectra for lowest fecal concentration (9%) with highest fecal concentration (34%); specifically looking at signals for proline and glucose – two compounds which had profiling confidence sensitive to fecal concentration. Metabolite signals highlighted in red represent concentrations profiled to fit the low fecal concentration spectrum. Light grey line represents the subtraction line between the 9% fecal water spectrum and all profiled metabolites. Note: only the proline and glucose NMR signal clusters found within 3.2-3.5 ppm are presented here.

5.1.3 Discussion

5.1.3.1 Trends in Metabolite Profiles

The metabolite profiles followed a linear trend as fecal concentrations decreased, as expected. Ethanol was the only compound that did not have a strong, linear relationship between compound concentration and fecal concentration. Currently the reason for such a randomized pattern of ethanol concentration observed is unclear. While vapour pressure could be a contributing factor, other volatile compounds, such as methanol, which has a higher vapour pressure than ethanol[128], did not exhibit random patterns.

5.1.3.2 Normalization of Metabolite Concentrations

The metabolite concentrations were normalized to the amount of feces present in the samples by dividing the metabolite concentrations by the concentration of feces used to prepare the samples. The normalized compound concentrations were calculated to be very similar across samples of various fecal concentrations, re-enforcing the linear relationship between the two parameters. While normalizing the metabolite compounds by fecal concentration enables one to compare samples prepared from the same stool sample, it does not address the variability in water content from one stool sample to the next. Not accounting for the water content of the original stool sample leaves dilution as a factor that is not fully controlled in preparation of fecal water samples, and could ultimately skew the metabolite measurements. One means of overcoming this challenge is to estimate the water content of feces by weighing the stool sample before and after freeze drying[129]. Therefore, a true means of normalizing metabolite measurements to fecal matter content would account for dilutions made during the fecal water preparation process as well as dilution due to the water content of the original stool sample.

Alternatively, reporting the metabolite concentrations independent of feces content circumvents measurement of water content and fecal water dilutions. Normalizing the metabolite profile to the most prominent compound, acetate, presented the metabolites independent of fecal concentrations, while still able to make comparisons across several fecal donations. Acetate is an attractive normalization marker as it is one of the main microbial fermentation products and has been consistently reported as one of the most abundant compound found in fecal metabolomes[3,61,130]. Even in *in vitro* conditions designed to favour production of other SCFAs, acetate is still present in a relatively significant amount[131]. In performing this normalization, the acetate-normalized compound concentrations had small standard deviations, which corresponded to the linear regression model for compound concentration vs fecal concentration. In fact, compared to normalization by fecal concentration, acetate-normalization yielded a narrower spread of normalized values obtained from samples varying in amount of feces content. It should be noted that since acetate-normalization expresses compounds as a ratio, evaluations on acetate levels and comparisons of metabolite levels based on magnitudes are masked. Therefore, it would be ideal to report on both the compound concentrations, as well as their ratio values.

5.1.3.3 Impact of Spectral Limitations

The “profile-ability” of the NMR spectra of fecal water samples should also be taken into consideration when choosing the appropriate fecal concentration. As the fecal concentration decreases, the signal-to-noise ratio of low-concentration compounds worsens and the accuracy and confidence at which the compound can be quantified and identified deteriorates. Proline, choline, pyroglutamate, p-cresol, arabinose and glucose were highlighted as compounds with which the profiling confidence was sensitive to low fecal concentrations. Even though they were not necessarily the compounds found at the lowest concentration in the profiles, they had lower thresholds for

limitations of quantification and identification due to their chemical shift, surrounding peak signals, and signal-to-noise ratios of peaks. Essentially, at low fecal concentrations, the metabolite concentrations of certain compounds were no longer within the limit of quantification, and in the case of arabinose, within the limit of detection. These profiling concerns pertain mostly to the fecal water samples of 9% and 19% fecal concentration. In comparing the samples containing fecal water within the range of 25%-34% fecal concentration, the higher fecal concentration samples did not differ greatly from the mid-range fecal concentration, in terms of ability to distinguish compound signals from noise signals.

5.1.3.4 Minimum Level of Feces Required in Fecal Water Samples

As mentioned earlier, the composition of fecal samples can be highly variable; for example, a gram of one stool donation could have more water content than one gram of another stool donation. Instead of evaluating a fecal water sample based on its concentration of feces, the sample could be identified based on an internal compound. Acetate is a compound that has been consistently observed at high concentrations in fecal water samples, and could be used not only to normalize the compound concentrations, but also to gage the contents of the fecal water sample. The samples of decreasing fecal concentration demonstrated that fecal concentrations at 25% and higher had metabolite profiles that sufficiently cleared the limit of quantification and limit of detection for all the compounds. This minimal fecal concentration could also be expressed as 7.3 mM of acetate, or 0.43 g/L of acetate. The linear relationship between fecal concentration and acetate concentration as well as the ubiquity of acetate in fecal water sample enables this compound to act as a proxy in evaluating the contents of fecal water samples.

Chapter 6 Metabolomic Characterization of Gut-Mimicking CSTR

6.1 Chapter Objective

The purpose of this chapter is to use a metabolomic approach to evaluate the *in vitro* tools used to study the human gut microbiota. Two experiments were performed with the aim of establishing further methodological understanding of both fecal water samples and community cultures grown *in vitro* in an anaerobic continuous stirred tank reactor (CSTR). The first experiment focused on developing a protocol for preparing samples of gut microbiota grown *in vitro* for NMR-based metabolomic analysis purposes. In determining the effects of different stages of sample preparation, such as syringe filtration, centrifugation, etc., a protocol for NMR-based metabolomic analysis was developed. The second experiment presented in this chapter addresses the limited degree standardization of the fecal inocula used for *in vitro* community cultures. Here, the impact of the inoculum on the resulting *in vitro* culture was investigated by culturing several defined communities derived from inocula that contained the same bacterial species, but varied in the proportions at which the bacteria they were present in the inoculum.

6.2 Gut-mimicking CSTR Effluent Sample Preparation and Analysis

6.2.1 Experiment Objective

Samples collected from the gut-mimicking bioreactors require an initial processing step to remove cell material and large media particulates before being suitable for metabolite analysis. The purpose of the following experiment was to determine the effects of various preparation protocols on the metabolite profiles. Two techniques – ultracentrifugation or syringe filtration – were examined to establish an appropriate sample preparation protocol.

6.2.2 Materials and Methods

6.2.2.1 Experiment Design

Along with testing five different ultracentrifugation speeds, two filtration procedures were also tested to see if they could be used as an alternative method for cell and particulate separation. The experiment was conducted to evaluate the effects of various stages in the sample processing procedure. In a first step, some samples were subjected to a "soft spin" (4000 rpm 4, °C, 3 min) to remove the bulk of large insoluble material (pre-processing stage) before being subjected to higher centrifugal forces. In a second step (processing stage), samples were subjected to either ultracentrifugation (4 °C, 2 h) or syringe filtration (35 mm). Following these two stages, all samples were subjected to a post-processing stage consisting of filtration, through a 0.22 μm syringe filter. The different conditions examined are outlined in Table 6-1.

Table 6-1. Sample treatment conditions.

Pre-Process	Process		Number of Replicates	
Soft spin	**Ultracentrifuge**		**Experiment 1**	**Experiment 2**
	rpm	**g**		
Yes	10 000	7 360	3	3
Yes	20 000	29 430	3	3
Yes	30 000	66 230	3	3
Yes	40 000	117 730	3	3
Yes	50 000	183 960	3	3
No	10 000	7 360	3	0
No	50 000	183 960	3	0
Soft spin	**Syringe filtration**			
Yes	Serial filtration (1.0µm, 0.80µm, 0.45µm)		3	0
Yes	Single-step filtration		3	8

6.2.2.2 Sample Collection and Handling

An effluent reservoir was collected and stored at 4 °C on the 28th day post-inoculation from a CSTR system propagating feces-derived cultures obtained from Donor A. This reservoir was mixed with a stir bar and sampled in 30-mL volumes for subsequent processing. The effluent reservoir ensured that variability seen in the metabolite results originated from processing differences, and not due to sample-to-sample variability. Ultracentrifugation was performed on Beckman L8-55M, with a Beckman Ti70 rotor and 30-mL Beckman polyallomer Optiseal tubes. Once processed, samples were scanned by NMR.

6.2.2.3 Statistical Analysis

The precision of the metabolite profiles was assessed by analyzing the relative standard deviation (RSD) of the control, non-ultracentrifuged samples in Experiment 2 (Appendix B1.). The two types of filtration procedure, serial and single-step filtration, were compared to each other using Student's t test at the 95% confidence level. A linear regression was used to model this data from all the processing methods (serial filtration, single-step filtration and ultracentrifugation), and the significance of this model was tested by ANOVA. Finally, the metabolite profiles from both experiments were modelled to determine which factors affected the trends observed. This was done using a multiple linear regression (MLR) model using the equation shown in Appendix B2. Since the majority of the compounds demonstrated linear trends, the non-linear terms in this model were not evaluated. Furthermore, the experiment-process interaction term was not tested since none of the samples in Experiment 2 were subjected to the soft spin pre-processing step. The three-factor interaction term was also not evaluated since the small number of replicates would be unlikely to yield enough information to assess such an interaction. The significance of each term in this complex model was evaluated at $p \leq 0.0009$, which is the alpha value of 0.05 after it has been subjected to the

Bonferroni correction (Appendix B3.) to reduce sensitivity to type I error. In this way, a more simplified model was determined. The simplified model produced by each iteration of model simplification was compared to the previous model by ANOVA to assess whether the two models were statistically different. The simplified model comments on the effects of ultracentrifuge speed, experiment, and the pre-processing step on the metabolite profiles. The effects of ultracentrifuge speed was evaluated using the percent change in concentration per 10 000 rpm for each metabolite. This was obtained by using the slope and predicted concentrations from the simplest MLR model (Appendix B4.). The absolute percent decrease in concentration per 10 000 rpm was also analyzed as a function of molecular weight of the corresponding compound. The importance of bias effects was determined as the absolute percentage of the bias coefficient divided by the intercept of the simplified MLR model.

6.2.3 Results

6.2.3.1 Profiling Variability

The samples subjected to single-step syringe filtration in Table 6-1 were used to assess the precision of the metabolite. Since the eight replicate samples were aliquots of the one sample that had been subjected to the preparation protocol, the only source of variability would be the steps involved in constructing the metabolite profiles. The precision of each compound was evaluated based on its relative standard deviation, where the acceptable level of precision was set at 15%, or if near the limit of quantification, 20%. The only compounds that had a RSD value greater than 15% were p-cresol (20.6%) and thymine (16.7%) (Figure A4). Indeed, both p-cresol and thymine were near their limits of quantification in the samples assessed for precision.

6.2.3.2 Processing Methods

Ultracentrifugation speeds, serial filtration (1 μm, 0.8 μm, 0.45 μm and 0.22 μm) and single-step filtration (0.22 μm only) were tested. One of the samples subjected to single-step filtration in Experiment 2 had an unusually high concentration of formate (>0.3 mM), whereas all concentrations of formate in other filtered samples, single-step or serial, did not exceed 0.13 mM. Since this point was an outlier, it was excluded from all analyses. Between the serial filtration and single-step filtration processing methods, no compounds were found to be statistically different. Since the two filtration process methods did not create significantly different metabolite profiles, they were grouped as one set of "control" samples (0 rpm) for the ultracentrifuge-based processing methods.

6.2.3.3 Sources of Effects Impacting Metabolite Concentration

The metabolite profiles were modelled to describe the observed trends in compound concentrations. This model was formed by eliminating non-significant terms from a complex, general model until the simplest model that described the relationships remained. As seen in Figure A3, no compounds in the two interaction effects and the pre-processing effect were significant, revealing that these factors did not contribute to the concentration trends observed, and thus should be removed from the model. The simplest model illustrated that the trends observed were the effects of ultracentrifuge speed and experiment only. This indicated that the effect of the ultracentrifuge speed, which was significant for almost every compound, was independent of other factors, such as experiment and preprocessing steps. Similarly, the effect of experiment, where it was significant, was also independent of all other explanatory factors.

6.2.3.4 Effect of Ultracentrifuge Speed

The simplest MLR model that described the metabolite data showed that almost all compounds in the metabolite profiles demonstrated significant linear relationships with ultracentrifuge speed. These compounds demonstrated a negative linear relationship between compound concentration and ultracentrifuge speed, except for p-cresol, which had a positive linear relationship in both experiments (Figure 6-1). The compounds that did not exhibit significant linear changes with increasing centrifugal force are shown in Table A4. While the linear relationships exhibited by most of the compounds were statistically significant, the percent decrease in metabolite concentration per 10 000 rpm was on average 1.8% $\pm$ 0.87% in Experiment 1 and 1.8% + 0.90% in Experiment 2. These linear decreases in concentration with ultracentrifuge speed appeared to be directly related to the molecular weight of the corresponding compound in Experiment 1. However, this effect was not reproduced in Experiment 2, where no significant relationship was observed between the absolute percent decrease and molecular weight (Figure 6-2). The only compound to increase in concentration with centrifugal force was p-cresol, which exhibited a 49.5% increase per 10 000 rpm in Experiment 1 and 41.3% increase per 10 000 rpm in Experiment 2.

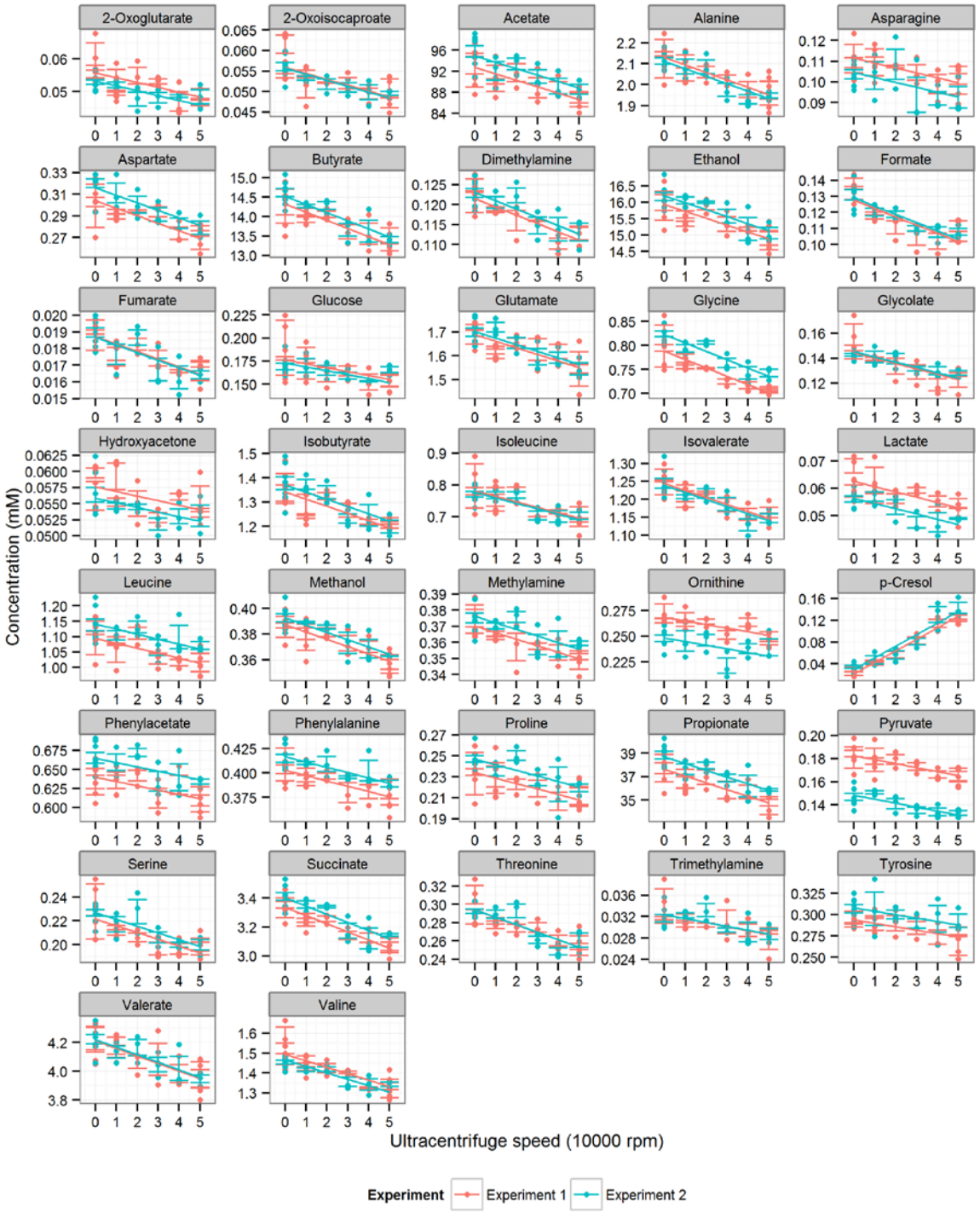

Figure 6-1. Compounds that demonstrate significant linear relationship with centrifugal force.

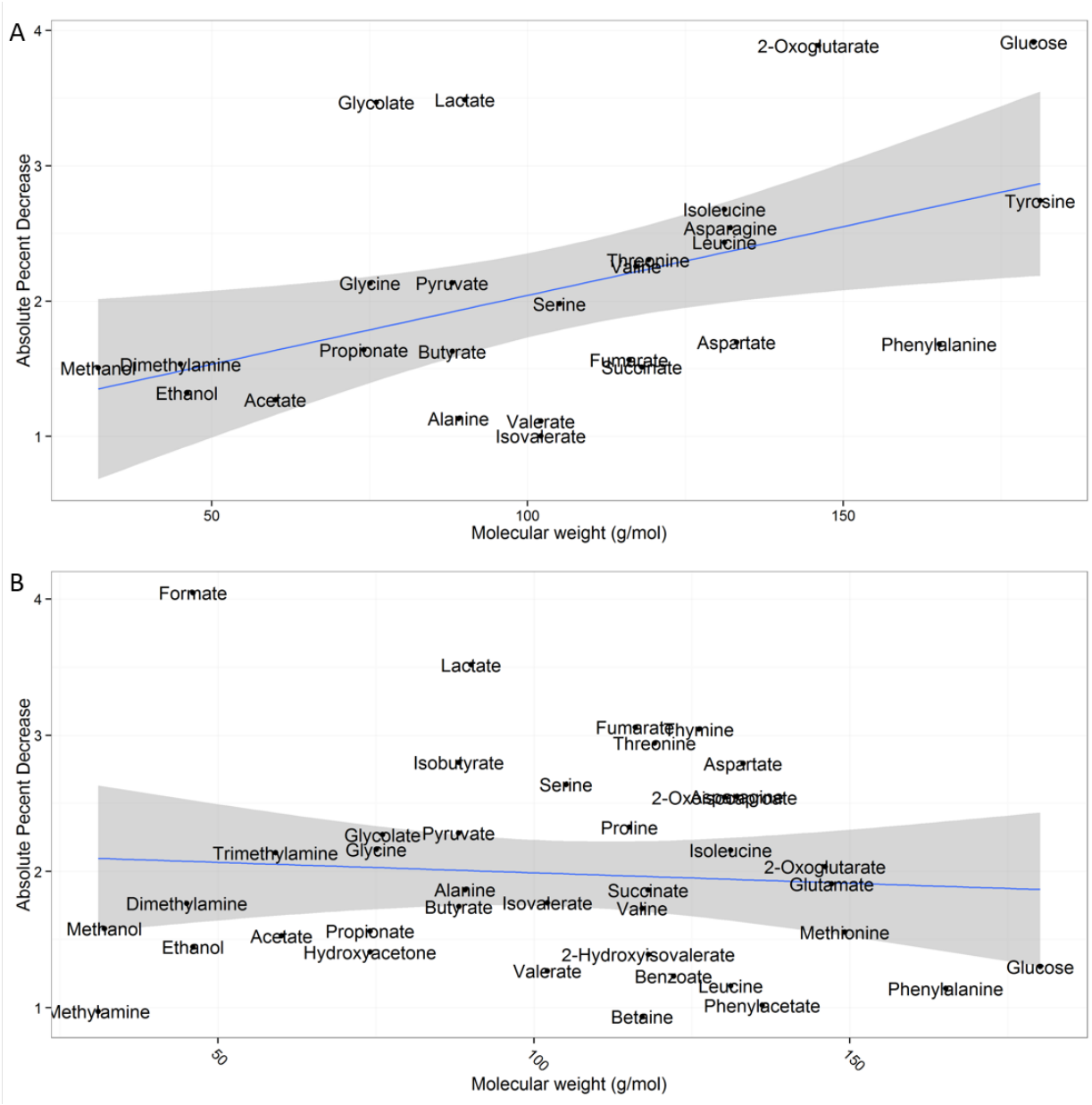

Figure 6-2. Effect of molecular weight on percent concentration decrease. (A) Experiment 1, y = 21.86x + 59.65, p-value = 0.015; (B) Experiment 2, y = -3.84x + 110.76, p-value=0.641.

6.2.3.5 Comparison of Replicate Experiments

52 compounds were identified and quantified in the metabolite profiles in both experiment replicates. The same compounds were found in the metabolite profiles of both experiment sets. An overview of the two experiments indicated that the two sample sets were very similar to each other, as overlaying the metabolite profiles from the two experiment's samples revealed that the compound concentrations followed similar trends (Figure 6-1). In conducting the experiment twice, the degree to which the ultracentrifuge effects were reproduced from one experiment to the next was assessed. 38 compounds exhibited the same trend in both experiments, in both the concentrations observed and the change in concentrations observed (Table A5). In some compounds however, the concentrations observed were biased by the experiment but the rate of concentration change remained the same between Experiment 1 and Experiment 2 (Figure 6-3). There was no consistency across these 14 compounds which had greater concentrations in the second experiment. Furthermore, five of these compounds, benzoate, beta-alanine, betaine, pyroglutamate and uracil, did not have significant changes in concentration with ultracentrifuge speed in either experiment. Even though the experiment bias observed in these compounds were significant, the importance of this effect was revealed when the magnitude of the experiment difference was examined relative to the concentrations observed (Table 6-2). Most compounds with significant bias only exhibited concentration differences between 2-10% of the observed concentration. Uracil, beta-alanine and pyruvate changed by 11 $\pm$ 2 %, 15 $\pm$ 2 % and 19 $\pm$ 1 %, respectively, and of these three, only pyruvate demonstrated a significant decrease with ultracentrifugation.

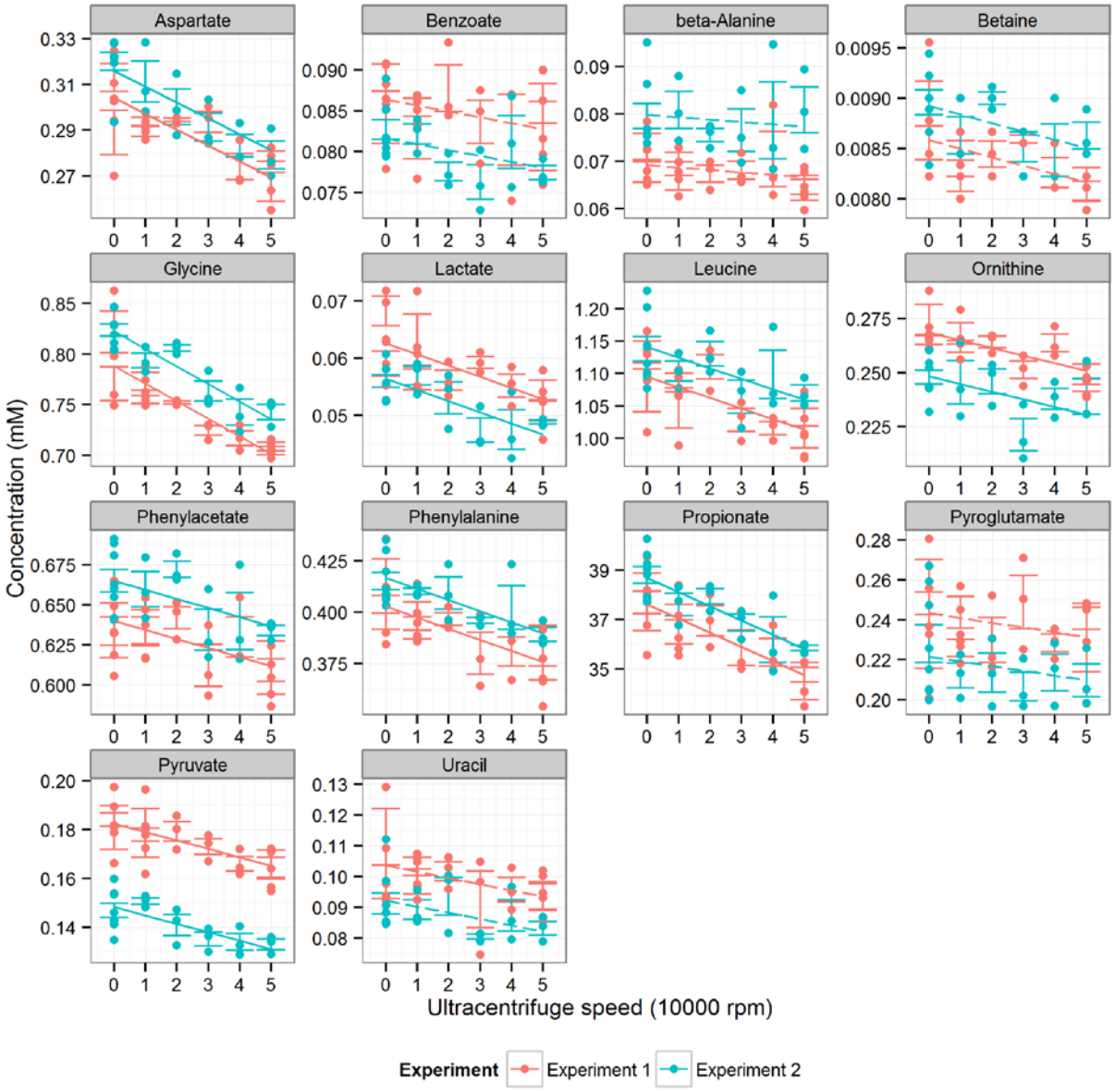

Figure 6-3. Compounds that had significant experiment bias based on MLR model. Slopes significantly different than zero shown as solid lines, otherwise, depicted as dashed lines.

Table 6-2. Absolute percent change due to experiment bias based on: $y = \beta_{speed}x_{speed} + \beta_{experiment}x_{experiment} + intercept.$

Compound	Intercept	Experiment Bias	Bias Standard Error	Absolute Percent Change	Standard Error of Percent Change
Aspartate	0.30	0.012	0.003	3.9	1.1
Benzoate	0.09	-0.005	0.001	5.5	1.4
beta-Alanine	0.07	0.011	0.002	15.4	2.6
Betaine	0.009	0.0003	8.88E-05	4.0	1.0
Glycine	0.79	0.0347	0.005	4.3	0.7
Lactate	0.06	-0.006	0.001	9.0	2.0
Leucine	1.1	0.046	0.012	4.2	1.1
Ornithine	0.27	-0.02	0.003	7.5	1.1
Phenylacetate	0.64	0.025	0.005	3.9	0.9
Phenylalanine	0.40	0.014	0.004	3.5	0.9
Propionate	37.6	1.1	0.27	2.8	0.7
Pyroglutamate	0.24	-0.02	0.006	8.9	2.3
Pyruvate	0.18	-0.03	0.002	18.7	1.2
Uracil	0.10	-0.01	0.002	11.0	2.4

6.2.4 Discussion

6.2.4.1 Profiling Variability

The metabolite profiles obtained in these experiments can be relied upon to provide precise values. The eight samples subjected to single-step filtration in Experiment 2 were used to evaluate the variability of each compound concentration. All of the compounds demonstrated good precision in their quantification, with RSD values less than 15%. The only two compounds that exhibited greater spread than that were p-cresol and thymine, which were near their limits of quantification. These guidelines for acceptable thresholds of precision are outlined in bioanalytics methods validation protocols from the American Food and Drug Administration (FDA)[132].

6.2.4.2 Pre-Processing Treatments

One of the elements tested involved the presence or absence of a "soft spin" step prior to any ultracentrifugation. The purpose of the "soft spin" was to remove cells and cell debris without exposing the supernatant to intracellular metabolites. This step was excluded for two ultracentrifugation conditions in order to isolate the effects of the "soft spin." When the metabolite concentrations from duplicate experiments were evaluated as a function of all the experiment factors (ultracentrifuge speed, pre-processing treatment and experiment replicate) and their possible interactions, pre-processing treatment did not influence the concentration observed for any of the compounds. Therefore, the pre-processing step of performing a soft spin prior to ultracentrifugation did not affect the metabolite concentrations.

6.2.4.3 Effect of Ultracentrifuge Speed

Negative linear relationships between compound concentration and centrifugal force applied to the samples were consistently observed throughout the metabolite profiles, prevalent to 37 of the 52

compounds observed. While the relationship between compound concentration and centrifugal force is clear, the absolute rate of concentration change does not seem to be a function of the molecular weight of the compounds. Furthermore, the concentration changes observed were minimal, averaging ~2% decrease in concentration per 10 000 rpm. The only compound found to increase linearly with centrifugal force was p-cresol, averaging at 45.4% per 10 000 rpm. Given the high rate of increase with centrifugal force applied, compounded with the contextual relevance this compound has to enteric microbial metabolism, it is evident that importance should be placed on being transparent regarding sample preparation procedures. Anaerobic catabolism of aromatic amino acids, phenylalanine, tyrosine and tryptophan, occur at the aliphatic side chains to produce phenols like p-cresol. p-Cresol has been shown to be produced by *Clostridium difficile* and *C. scatologenes* from tyrosine via decarboxylation of 4-hydroxyphenylacetate[133,134]. Furthermore, p-cresol was formed from cell-free extracts of *C. difficile*, where decarboxylation of 4-hydroxyphenylacetate was facilitated by combining the protein fraction of the cell-free extracts with the filtrate fraction, serine, or an α-keto acid like pyruvate[134]. The proposed mechanism for converting p-cresol from 4-hydroxyphenylacetate was a two-protein system where a decarboxylase was kept in the active, reduced form by an α-keto acid dehydrogenase. While 4-hydroxyphenylacetate was not observed in the samples, it has been observed in the metabolite profiles of other feces-derived cultures within this body of work. Therefore, it is possible that the level of 4-hydroxyphenylacetate in this sample set was below the level of detection. Upon examining the absolute rate of concentration change with ultracentrifuge speed, the combined rate of tyrosine and phenylalanine was still ~2 times lower than that of p-cresol, suggesting that if tyrosine and phenylalanine does play a role, they cannot be the only source. Tryptophan was not observed in the metabolite profiles.

6.2.4.4 Comparison of Replicate Experiments

Samples obtained from the same pool of bioreactor effluent were used in both replicate experiments to eliminate sample-to-sample variation. Indeed the compounds that were identified in the metabolite profile from one replicate experiment to the next were identical. The concentration changes observed in the duplicate experiments were explained independently by ultracentrifuge speed and experiment. This is important because an interaction effect between these two factors, indicated by different rates of change in concentration depending on the experiment, would infer that compounds were inconsistent in their response to ultracentrifugation from one experiment to another, even under controlled conditions. The linear regression modelled above demonstrated that there was no significant interaction occurring between any of the experimental factors, so the metabolites' response to the sample preparation protocols was consistent, and thus, predictable. Experiment bias, indicated by a parallel shift in the linear regression from one experiment to another, would mean that there were overall concentration differences between experiment sample sets. Of the 14 compounds that had significant experiment bias, most of these compounds differed by only 2-10% of the observed corresponding compound concentration. Uracil, and beta-alanine had greater discrepancies between experiments, revealing that these two compounds may be sensitive to factors not controlled within this experiment. Pyruvate levels were also not reproduced from one experiment to the next, as it also demonstrated significant experiment bias. However, the impact of ultracentrifuge speed on pyruvate levels was reproduced, with levels decreasing by 2% per 10 000 rpm in both replicate experiments. In general, the compound concentrations between the two sample sets were identical and the compounds response to ultracentrifugation could be predicted.

6.3 Effects of Varying Inoculum Biomass Proportions

6.3.1 Experiment Objectives

The objective of this experiment was to determine the impact of the biomass makeup of inoculants on the resulting *in vitro* community. To evaluate this, defined communities were cultured from inocula that contained the same bacterial species, but the proportional makeup of the bacteria in the seeding culture varied across inocula. The hypothesis driving this experiment was that *in vitro* communities develop similar metabolite profiles when the seeding culture is composed of the same bacterial strains, regardless of the proportions at which the bacteria are present in the inoculum.

6.3.2 Materials and Methods

6.3.2.1 Experiment Outline

Two different defined communities, Microbial Ecosystem Therapeutic-2A (MET-2A) and MET-3A were studied in a CSTR system that consisted of twin vessels, operating in parallel. In a given CSTR run, one vessel was inoculated with a defined community and the other was inoculated with the same defined community, but with different biomass proportions of the same bacteria strains. The varied communities were termed MET-2B or MET-3B, depending on the respective parent community. Culturing each defined community and its reciprocal community was repeated 5 months after the initial experiment (Figure 6-4). Therefore, Run 1 and 3 are duplicate experiments of each other, and Run 2 and 4 are duplicate experiments.

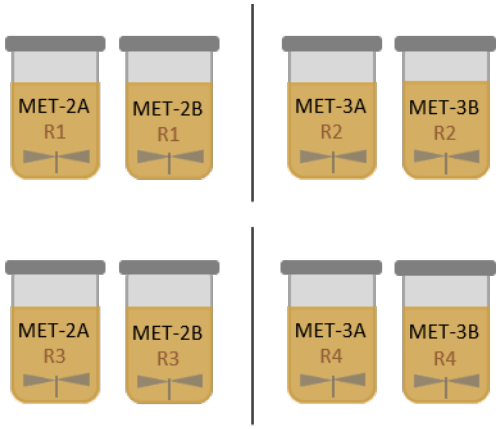

Figure 6-4. Experiment overview. Variants of two different defined communities derived from fecal communities obtained from different donors were examined: MET-2 and MET-3. The A and B variants of the MET-2 and MET-3 communities have the same bacterial strains, but have varied proportional makeup of those bacteria. Run 1 (R1) and Run 3 (R3) were duplicate experiments and Run 2 (R2) and Run 4 (R4) are duplicate experiments.

6.3.2.2 Sampling and Storage

Samples were collected directly from the bioreactor vessel throughout the duration of the bioreactor runs to monitor the cultures' growth. Long, sterile sampling needles that punctured a silicone septum were used to directly access the bacterial culture in the bioreactor vessel. 4-mL samples were aliquoted into 2 2-mL tubes and stored at -80 °C. Samples from day 6, 14, and 20 from each bioreactor run were selected for metabolite analysis. Samples were thawed, then centrifuged at 14 500 rpm for 5 min to pellet the cell fraction. The supernatant was subsequently filtered through 0.45 μm and 0.22 μm syringe filters (Corning; Corning, New York) and stored for up to 4 days at 4 ºC until NMR scanning.

6.3.2.3 Statistical Analysis

The metabolite profiles of each community culture were annotated by identifying the compounds' chemical taxonomic classes. The metabolite profiles could then by analyzed according to these taxonomic classes, examining the prevalence of each compound class within a profile, and how these class proportions change when weighted to compound concentrations. The chemical taxonomic class of each compound was identified based on classifications listed in the Human Metabolome Database[135] (www.hmdb.com). The magnitude of the metabolite profile represented by a given compound class, referred to as the class weight, was calculated as the sum of concentrations of metabolites of that class within a metabolite profile. The class weight as a proportion, referred to as class percentage, was presented by dividing the class weight by the total sum of metabolite concentrations in the profile. The mean class weight within a given culture was also calculated. Two samples were selected at random to be scanned in triplicate and be used to assess profiling variability. The precision of the compound concentrations were assessed based on the relative standard deviation (RSD) in each triplicate set. RSD was calculated as shown in Appendix B1. The mean of each set of

triplicates was used as the concentration for each respective sample (that were analyzed in triplicate). Student's t tests were also applied to assess the run-to-run differences between the two pairs of duplicate experiments performed for each defined culture, MET-2A, MET-2B, MET-3A and MET-3B. It should be noted that the three time-course data points from each run were considered as one group when comparing the duplicate runs. While a time-matched comparison across duplicate runs would be more ideal, comparing the runs across all three time points was sufficient as the two runs were expected to behave the same. Even so, a more stringent level of significance would give more allowances for some dynamicity in the progression of the runs. Moreover, performing the Student's t test repeatedly (for every compound and every defined culture) leaves the test to be prone to type I error. For these two reasons, a Bonferroni correction was applied to the alpha value, which defined the significance level as $p \leq 0.0075$ (Appendix B3.). PCA was used to analyze the metabolite profiles. This multivariate analysis was performed using the function "prcomp" within the R base package "stats". Linear regression was used to assess the trends of compound concentrations over time (day 4-20) via the "aov" function in the R base package "stats". This function is a form of ANOVA that assesses the variation of concentrations about a linear regression model and compared it to the variation of the concentrations about the grand mean. Significance was determined at the 90% confidence level ($p \leq 0.1$). This confidence level was chosen because in this case, several weak linear trends had significance values between 0.05 and 0.1.

6.3.3 Results

6.3.3.1 Profiling Variability

For the majority of compounds, the relative standard deviation was less than or equal to 15%. (Figure A5). Compounds that had a spread greater than 15% from the mean are shown in Table 6-3.

However, even with the spreads observed, the two samples done in triplicate could still be distinguished from each other as demonstrated by PCA (Figure 6-5).

Table 6-3. Compounds with RSD in replicate samples > 15%

Sample 1		Sample 2	
Compound	**RSD (%)**	**Compound**	**RSD (%)**
Betaine	16.5	Benzoate	19.3
Fructose	24.3	Fructose	21.4
Fumarate	21.3	Glucose	21.4
Hydroxyacetone	16.8	Glycine	28.0
Proline	28.4		

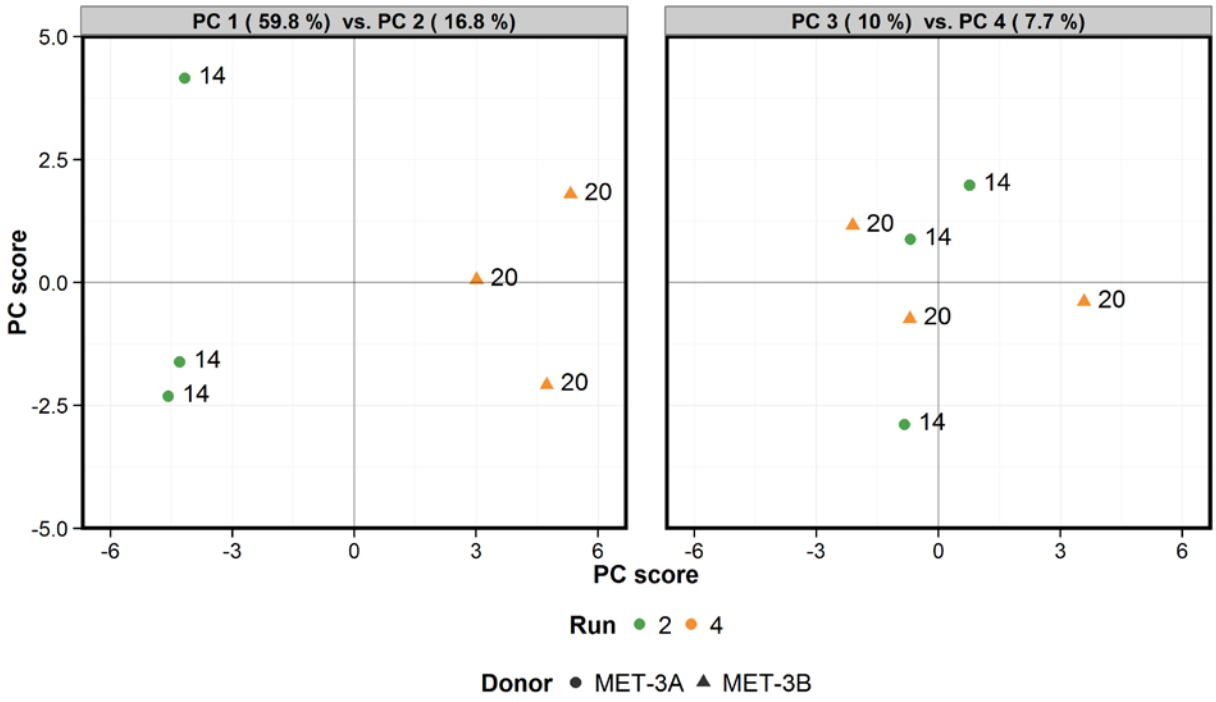

Figure 6-5. PCA score plot of triplicate samples.

6.3.3.2 Replicate Runs

Run 1 and Run 3 were duplicate experiments that propagated MET-2A and MET-2B. Run 2 and Run 4 were duplicate experiments that propagated MET-3A and MET-3B. While the duplicate Runs may have been identical in the seeding population, there were slight differences in the metabolite profiles of each pair of replicates (Figure A6). These differences became apparent in a PCA score plot of all the samples (Figure 6-6). In this overall model, Run 2 and 4 produced similar metabolite profiles of one another, but Run 1 and 3 clustered distinctly from one another. The variation along principal component (PC) 1, along the x-axis, accounted for 60.3% of the total variation and accounted for the differences between the defined culture types, MET-2A and MET-3A. The variation along the y-axis, PC2, occurred predominantly due to differences between Run 1 and Run 3 and differences between Run 2 and 4. Since differences between the duplicate runs occurred along PC2, which accounted for only 8.3% of the total variation, the differences between the duplicate runs were much smaller than the differences between the types of defined communities (MET-2A vs MET-3A).

Since all four cultures were propagated in duplicate runs, the reproducibility of each culture could be assessed by using Student's t-test to compare each compound between the two runs. The compounds that differed significantly between the duplicate runs are shown in Table 6-4. All of these compounds, except for fumarate, had within-run mean concentrations greater than 15% of the between-run mean concentration.

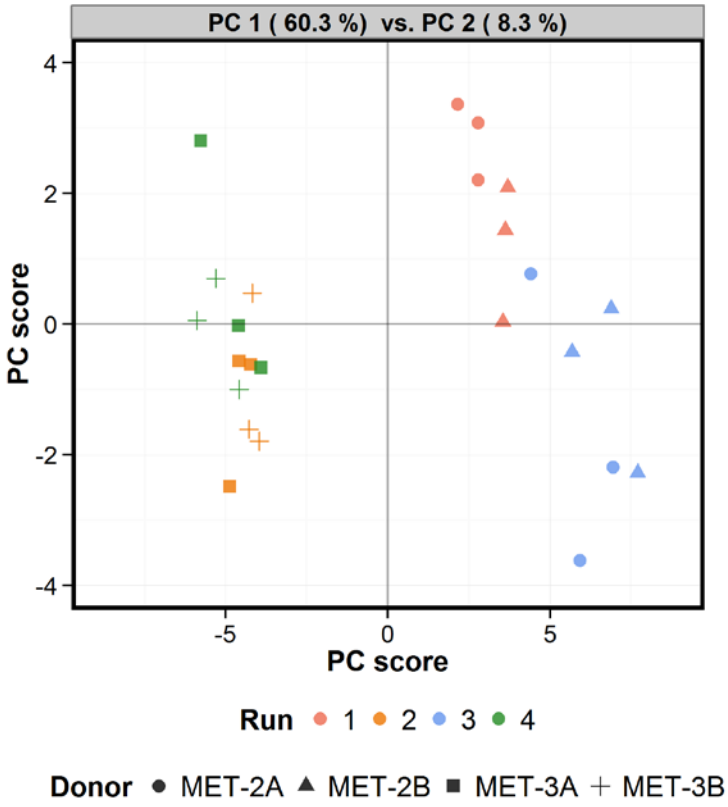

Figure 6-6. PCA score plot of metabolite profiles of all bioreactor samples. Metabolite concentrations have been mean centered and unit variance scaled in order for equal-weighted comparison between compounds.

Table 6-4. Compounds that differed significantly between duplicate runs.

Compound	Culture	T Statistic	P value	Absolute Percent Difference[3]
Fumarate	MET-2A	-5.1	0.008	8.8
Propionate	MET-2A	5.3	0.007	28.2
Aspartate	MET-2B	-10.7	0.002	28.5
Glycine	MET-3A	5.2	0.007	32.3
Glycine	MET-3B	5.5	0.005	43.9
Isoleucine	MET-3B	13.2	0.0003	16.5

[3]Percent difference = |mean concentration within a run – mean concentration from duplicate runs|/mean concentration from duplicate runs.

6.3.3.3 Metabolite Profiles

Metabolite profiles were constructed for 24 samples. Each profile contained 41 compounds where the assignment of the signal's compound identity and concentration were considered to be high confidence. The total concentration of the metabolite profiles was greater in MET-3A-based cultures (MET-3A and MET-3B) than to MET-2A-based cultures (MET-2A and MET-2B), and similar between a defined culture and its reciprocal (Figure 6-7). The structure of the metabolite profiles was fairly consistent across all four different seeding populations. Based on the number of compounds profiled, amino acid and amino acid derivatives (39%) comprised most of compounds identified, followed by fatty acids and conjugates, carboxylic acids and derivatives and monosaccharides which each accounted for ~10%. When the taxonomic classes in the metabolite profiles were weighted to the compounds' concentration, the proportions of classes were drastically altered. Carboxylic acids and derivatives dominated in the profiles, followed by fatty acids and conjugates, alcohols and polyols and amino acids and derivatives (Table 6-5). The distribution of these weighted compound classes was fairly similar across all four cultures, with slight differences between the MET-2A-type cultures and the MET-3A-type cultures. The most noticeable differences were a dramatic increase in carboxylic acids and a decrease in fatty acids in MET-3A-type cultures compared to MET-2A-type cultures.

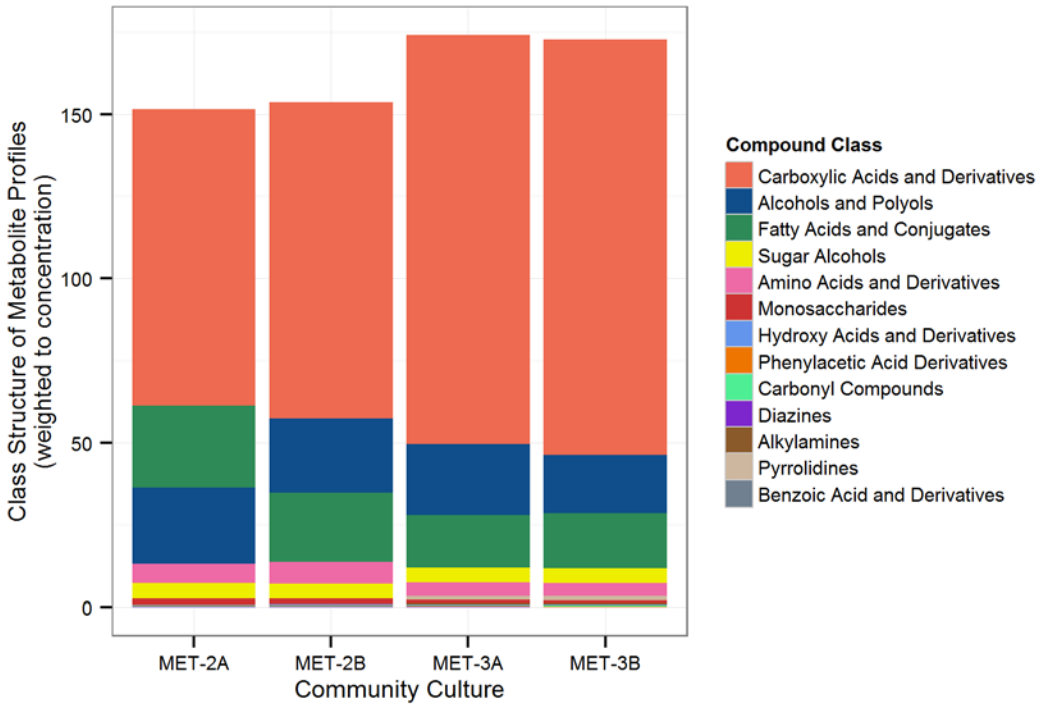

Figure 6-7. The breakdown of compound classes in metabolite profiles of each defined culture. Proportions of compound classes were weighted to compound concentration. The class weights shown are the mean class weights of profiles within a given culture.

Table 6-5. Proportion of Compound Classes, weighted to concentrations.

Compound Class	MET-2A	MET-2B	MET-3A	MET-3B
Carboxylic acids and derivatives	59.4	62.5	71.4	73.0
Fatty acids and conjugates	16.5	13.6	9.2	9.9
Alcohols and polyols	15.3	14.8	12.4	10.3
Amino acids and derivatives	3.9	4.4	2.3	2.4
Sugar alcohols	3.1	2.9	2.6	2.5
Monosaccharides	1.1	1.1	0.6	0.5
Hydroxy acids and derivatives	0.2	0.2	0.2	0.2
Phenylacetic acid derivatives	0.1	0.2	0.1	0.1
Carbonyl compounds	0.1	0.01	0.1	0.1
Diazines	0.09	0.01	0.1	0.1
Alkylamines	0.07	0.08	0.2	0.2
Benzoic acid and derivatives	0	0	0.01	0.01
Pyrrolidines	0	0	0.7	0.7

6.3.3.4 Defined Cultures vs Their Reciprocals

Most of the compounds tend to a common point over time between a given culture and it's reciprocal. Compounds that reach this common point at Day 20 in a weak linear fashion in at least one run are shown in Figure 6-8A and Figure 6-10A. The compounds shown in Figure 6-8B and Figure 6-10B reach a common point at Day 20, but in a non-linear pattern. It is notable that the achievement of a common point is not always observed across duplicate runs (Figure 6-9 and Figure 6-11). The compounds that do not reach a common point by Day 20 in any of the runs are listed in Table 6-6.

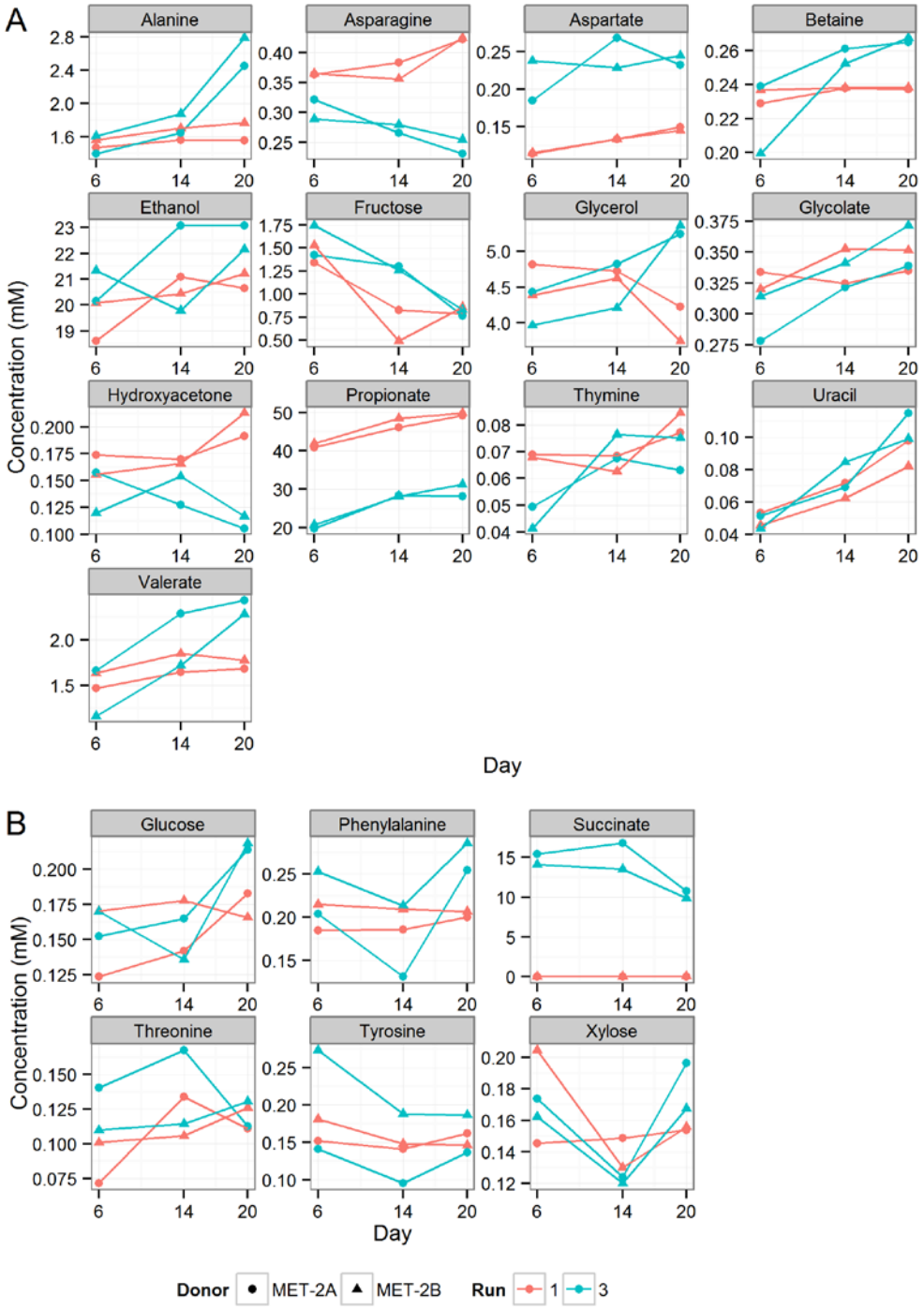

Figure 6-8. Compounds that reach the same level by Day 20 across a MET-2A culture and its reciprocal (MET-2B). (A) Compounds that display time trends that demonstrated a linear model in both runs. (B) Compounds that have time trends that are not linear.

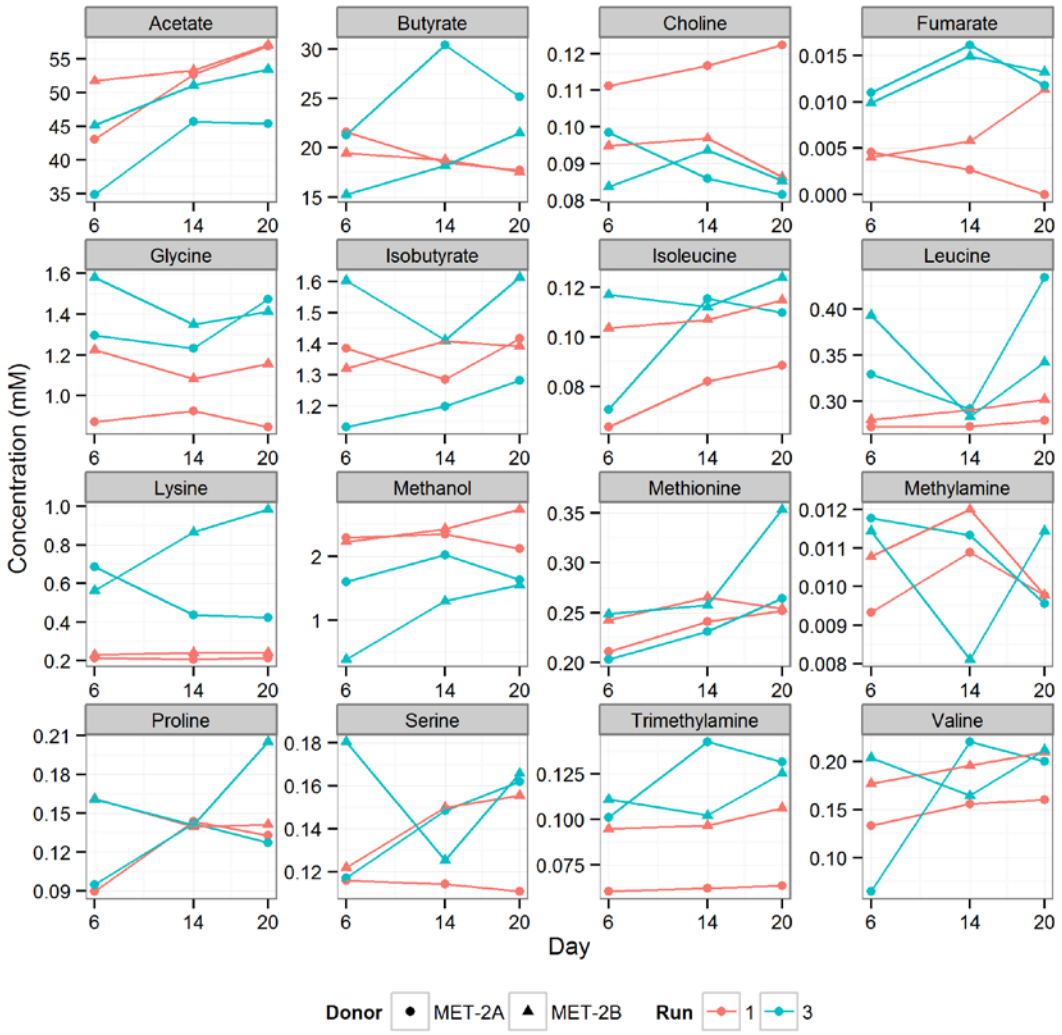

Figure 6-9. Common point reached in only one run for MET-2A and MET-2B.

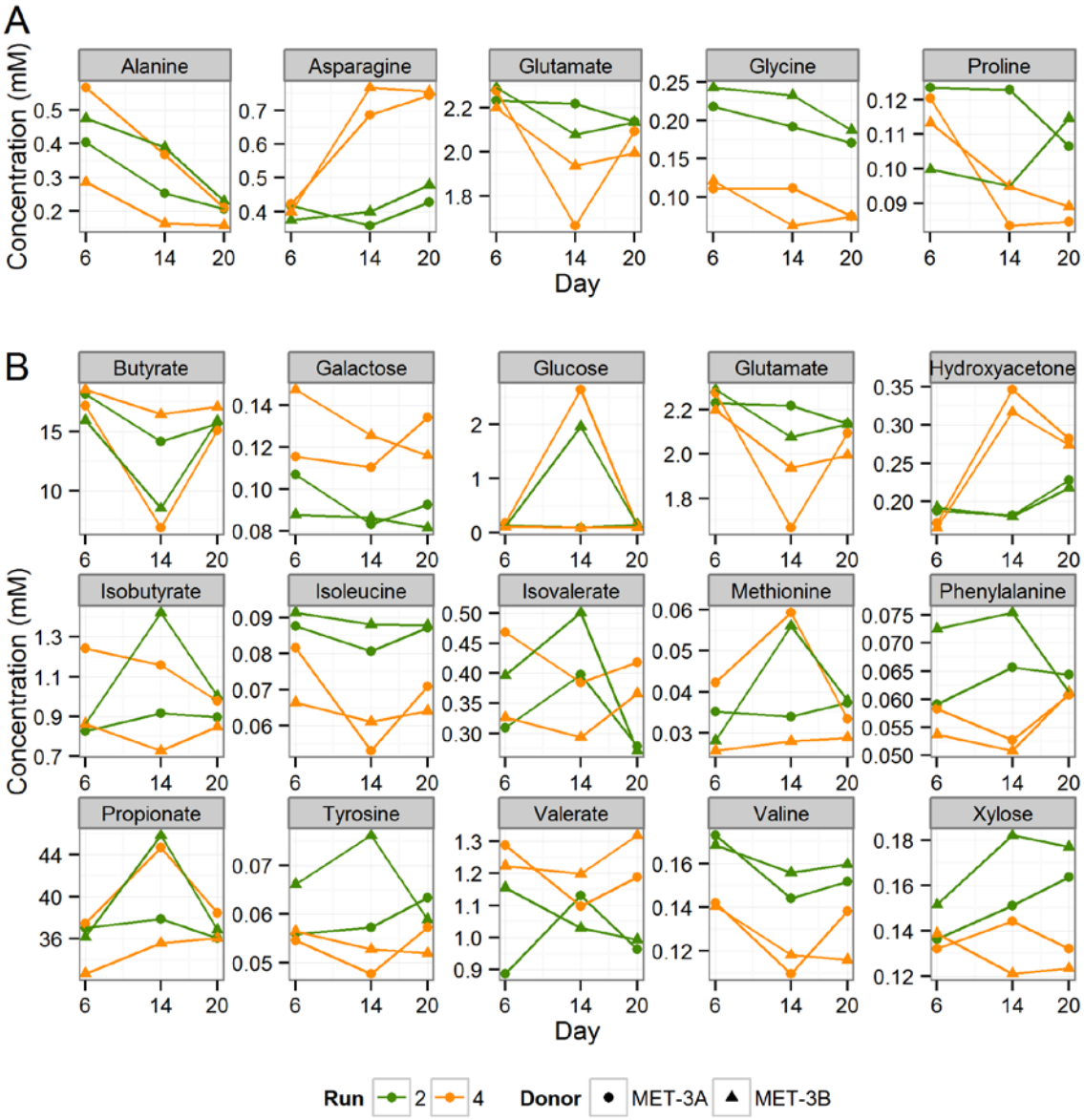

Figure 6-10. Compounds that reach the same level by Day 20 across a MET-3A culture and its reciprocal (MET-3B). (A) Compounds that display time trends that demonstrate a weak linear model in at least one of the runs. (B) Compounds that have time trends that are not linear.

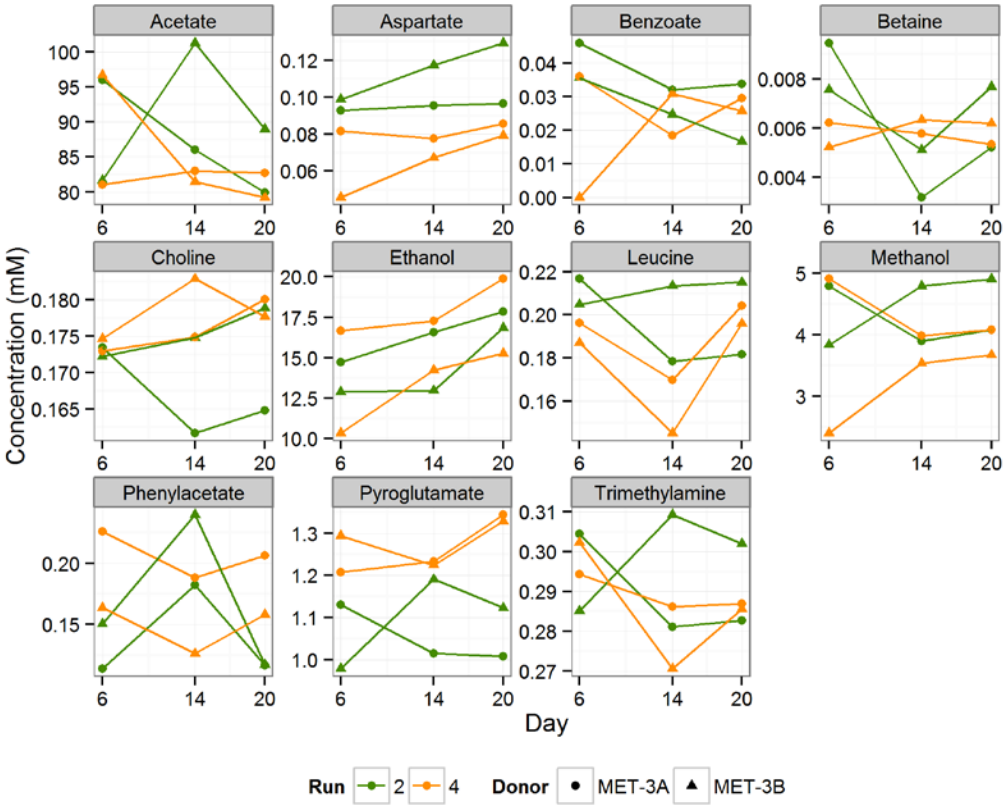

Figure 6-11. Common point reached in only one run for MET-3A and MET-3B

Table 6-6. Compounds that do not reach a common point between a culture and its reciprocal by Day 20 in either of the duplicate runs.

MET-2A vs MET-2B	MET-3A vs MET-3B
Galactose	Fructose
Isovalerate	Fumarate
Phenylacetate	Glycerol
	Glycolate
	Methanol
	Methylamine
	Serine
	Thymine

6.3.4 Discussion

6.3.4.1 Profiling Variability

Assessment of the metabolite profiles of replicate samples revealed that most compounds were quantified with a relative standard deviation of less than or equal to 15%. Since FDA regulations on bioanalytical method validation recommend 15% RSD, or 20% if at the lower limit of quantification, as the tolerance limit in assessments of precision[132], the metabolite profiles can be considered to have relatively little variation. The compounds that demonstrated more spread tended to be compounds with only a few number of clusters in its NMR signal, a higher degree of convolution with other spectral signals, or the compound peaks were masked due to poor signal-to-noise ratios. Even though these, or any combination of these spectral characteristics can increase the variation in quantification, the precision was still high enough to be easily distinguished from other samples.

6.3.4.2 Replicate Runs

The run-to-run effects comment on the repeatability of culturing the same bacterial population. The multivariate model that described the metabolite profiles explained that only 8.3% of the total variation could be attributed to run-to-run variability. Comparing the metabolite profiles of each of the duplicate runs revealed that most compounds levels were reproduced across runs, and only one or two compounds had significantly different concentrations from one replicate run to the next.

It should be noted that all time points from the growth of a culture were included in the run-to-run comparisons since only one data point was available for each time point. A more accurate evaluation of repeatability would compare runs only at time-matched points, where each time point had at least 3 data points. The implications of grouping all three time-points together in making run-run comparisons was that the test is more sensitive to compounds that remain constant over time

within one run. Compounds that fluctuate largely between day 4 and day 20 of a given CSTR culture would be more likely identified as an insignificant difference with other culture runs. That being said, the replicate runs can still be differentiated by gross concentration differences in the metabolite profiles.

6.3.4.3 Metabolite Profiles

The metabolite profiles of the four bacterial communities were very similar to one another, with the greatest degree of similarity between a culture and its respective reciprocal. The same 41 compounds were observed in the metabolite profiles in all four cultures. The most apparent differences in metabolite profiles were evident when comparing the profiles between MET-2A-based cultures and MET-3A-based cultures. Overall, the total concentration of compounds profiled was greater in MET-3-based cultures (MET-3A and MET-3B) compared to MET-2-based cultures (MET-2A and MET-2B). Furthermore, there was a larger proportion of carboxylic acids in the profiles of MET-3-based cultures (~72%) than in the MET-2-based cultures (~60%). This difference was mostly attributed to acetate, where the average concentration observed in MET-3A-based cultures was 86 $\pm$ 2 mM standard error, while the average concentration observed in MET-2A-based cultures was 49 $\pm$ 3 mM standard error. The comparatively low concentrations of fatty acids and conjugate compounds in MET-3-based cultures (~9% in MET-3-based cultures and ~15% in MET-2-based cultures) was mostly due to lower levels of butyrate. The mean concentration of butyrate in MET-3A-based cultures was 15 $\pm$ 1 mM standard error compared to 21 $\pm$ 2 mM standard error found in MET-2A-based cultures. Concentrations of valerate and isovalerate were also considerably lower in MET-3A-based cultures but they were found at fairly low concentrations in all four cultures, so they only accounted for ~1-2% of the metabolite profiles.

6.3.4.4 Defined Cultures vs Their Reciprocals

In general, there was very little distinction between each defined culture and their respective reciprocal culture. In fact, an overview of the time course progression of the compound changes indicated the cultures and their biomass-varied counterparts reached similar metabolite profiles by the 20^{th} day post-inoculation. For some compounds, this progression happened linearly, while in others, the compounds experienced some fluctuation before reaching a common concentration point at day 20. Only a select few compounds were not observed to reach a common point between a defined culture and its reciprocal cultures in either duplicate runs. While the convergence of compound concentrations by day 20 between a defined culture and its reciprocal culture was a first indication that the defined cultures tended towards a single metabolic steady state regardless of the biomass of species isolates in the inoculum, there are still aspects of this experiment that must be further investigated. First of all, it should be noted that the linearity was evaluated based on 3 time points of a given culture. For more robust evaluations on time series patterns, more time points should be evaluated, with multiple samples of each time point. This would enable more accurate modelling of the cultures' metabolic activities. Moreover, previous investigations reported that fecal cultures grown in CSTR systems do not stabilize in terms of species composition until ~5 weeks after inoculation[5]. Therefore, the cultures should be observed for longer in the each bioreactor run to observe when a true metabolic steady state is actually reached, and to determine if the similarities in the metabolite profiles of a culture and its reciprocal would maintained.

Chapter 7 Metabolomic Analysis of Human Fecal Microbiota Propagated *In Vitro* in a CSTR System

7.1 Chapter Objective

The purpose of this chapter was to demonstrate ways in which the metabolomic techniques and analyses used thus far can be applied to gain further understanding of the human gut microbiota. In the first experiment, metabolite profiles were used to describe feces-derived and defined *in vitro* communities. This study exemplified the potential of performing metabolomic analyses on *in vitro* gut microbial communities, as unique metabolic functions were identified in each bacterial culture. The second experiment supported the application of this metabolomic analysis for studying the human gut microbiota in a clinical and pharmaceutical framework. Several *in vitro* gut microbial communities were subjected to various types of perturbations, including antibiotics, a hormone and a defined community culture used to challenge the bioreactor culture. Examining the metabolomic responses to a wide range of perturbations set the preliminary groundwork in for raising attention to the metabolomic implications relevant to industrial and clinical research.

7.2 Metabolomic Analysis of the Human Gut Microbiota: Comparison of Feces-Derived Communities and Defined Mixed Communities

7.2.1 Journal Article Authorship

Sandi Yen[1], Julie A. K. McDonald[2], Kathleen Schroeter[2], Stanislav Sokolenko[1], Eric J. M. Blondeel[1], Emma Allen-Vercoe[2#], Marc G. Aucoin[1#]

[1]Waterloo Institute for Nanotechnology, Department of Chemical Engineering, University of Waterloo,

Waterloo, Ontario, Canada, N2L 3G1

[2]Department of Molecular and Cellular Biology, University of Guelph, Guelph, Ontario, Canada, N1G 2W1

[#]**corresponding authors:** Marc G. Aucoin maucoin@uwaterloo.ca, Emma Allen-Vercoe eav@uoguelph.ca

7.2.2 Justification of Original Work

With minor editorial changes to fulfill formatting requirements, this chapter is substantially as it has been submitted to the Journal of Proteome Research on October 31, 2014. The study was performed in collaboration with the Allen-Vercoe group at the University of Guelph. The partnership was formed where the Allen-Vercoe group brought forth expertise in culturing microbial communities representative of the human gut in an *in vitro* CSTR system and the Aucoin group extracted, analyzed and interpreted metabolomic data obtained from the microbial communities. Preparation and operation of the gut-mimicking CSTR system was performed by Julie McDonald and Kathleen Schroeter. They also collected raw fecal samples, prepared all community inocula, collected

bioreactor and effluent samples, and processed the samples in preparation for metabolomic analysis. 1D-^{1}H NMR spectroscopy was performed by Sandi Yen. The manuscript was written by Sandi Yen. This work is included in this thesis because it applied the skills and analytical techniques developed from earlier studies presented in this thesis.

7.2.3 Abstract

The extensive impact of the human gut microbiota on its human host calls for a need to understand the types of communication that occur amongst the bacteria and its host. A metabolomics approach can provide a snapshot of the microbe-microbe interactions occurring, as well as variations in the microbes from different hosts. In this study, metabolite profiles from an anaerobic continuous stirred-tank reactors (CSTR) system supporting the growth of several consortia of bacteria representative of the human gut were established and compared. Cell-free supernatant samples were analyzed by 1D-^{1}H Nuclear Magnetic Resonance (NMR) spectroscopy, producing spectra representative of the metabolic activity of a particular community at a given time. Using targeted profiling, specific metabolites were identified and quantified based on NMR analyses. Metabolite profiles discriminated each bacterial community examined, demonstrating that there are significant differences in the microbiota metabolome between each cultured community. We also found unique compounds that were identifying features of individual bacterial consortia. These findings are important because they demonstrate that metabolite profiles of gut microbial ecosystems can be constructed by targeted profiling of NMR spectra. Moreover, examination of these profiles sheds light on the type of microbes present in the gut and their metabolic interactions.

7.2.4 Introduction

In the wake of metagenomic research aiming to fully describe the gut microbiome, metabolomic research has emerged as a functional counterpart[136] that provides insight into the metabolic activities engaged by whole communities of microorganisms. Metabolomics is the study of the entire set of compounds, or metabolites, produced by a given organism, referred to as a metabolome. As a reflection of enzymatic activities and metabolic pathways carried out by the organism[137], the application of metabolomics to whole community bacterial cultures confers the ability to take a "snapshot" of the complex microbial population. Within the context of the human gut microbiota, the mixed culture metabolome not only reflects the metabolic activities of the enteric bacteria, but it also represents the bacterial compounds to which the human body is exposed.

Metabolomic studies of the gut microbiota often use metabolites from fecal extracts to compile metabolomes that capture the gut bacteria-host interaction[2,63,138]. The metabolomic data obtained directly from stool samples contains metabolites from both host and microbial activities. Thus, while fecal metabolomics provide insight into the gut microbial-host co-metabolism, it is not the ideal source for studying the independent metabolic contributions from either the host or enteric bacteria. Furthermore, the composition of the gut microbiota varies depending on host age, nutrition, behaviour and health[52,53,139]. Therefore, while fecal metabolomic studies bring forth understanding of the microbial-mammalian metabolic axis, the exact conditions are difficult to recreate to further investigate potential relationships. Taking the approach of microbial metabolomics to study the gut microbiota, analysis of the extracellular metabolites (the 'exometabolome') produced by the enteric bacteria circumvents host biology variability and focuses the metabolomic analysis on the community's metabolic state.

The robustness of using metabolomic approaches to study the gut microbiota relies heavily on the acquisition of high-quality data. The most common analytical platforms used in quantifying metabolomes are mass spectrometry (MS) and nuclear magnetic resonance (NMR) spectroscopy[140]. Both platforms have advantages and limitations that influence for which situations the analytical techniques are best suited. The methodology and application of MS and NMR in metabolomics have been extensively reviewed[23,140–142]. Within the scope of compiling enteric microbial metabolite profiles, ^{1}H NMR spectroscopy is the platform of choice because it requires minimal pre-processing of samples, enabling rapid yet reproducible spectroscopic results[29,39,143]. Its signals provide structural information about the compounds, facilitating identification of known compounds and classification of unknown compounds. Furthermore, targeted profiling techniques have been developed for deconvolution of NMR spectra and provide a level of quantification and identification that was not previously possible with binning algorithms, the traditional tool in analyzing NMR spectra[31,41].

In this study, we employed metabolomic characterization of feces-derived bacteria propagated as various mixed cultures in a continuous stirred tank reactor (CSTR) that mimicked the environment of the distal regions of the human gastrointestinal tract[5,144]. With this approach, we built quantitative metabolite profiles of different gut bacterial communities. The purpose of this study was to describe the metabolome of bacterial communities propagated in a CSTR, and to evaluate identifying characteristics of different populations. In deciphering the ecological dynamics of the gut microbiota, further understanding of the interaction between the gut microbial ecosystem and the human body can be achieved.

7.2.5 Materials and Methods

7.2.5.1 Experiment overview

The Research Ethics Board of the University of Guelph approved this study (REB#09AP011). This study was performed over 5 different bioreactor runs, where a run had one or more bioreactor vessels operating (Figure 7-1). All vessels within a run were inoculated from the same seeding sample. Conversely, each bioreactor run used different seeding sources, whether it was from a different donor, or from the same donor. Three healthy donors provided fresh fecal samples: donor A (male, 44 years-old), donor B (female, 26 years-old) and donor C (male, 25 years-old). None of these donors had a recent history of antibiotic treatment or gastrointestinal disorders within the 9 months prior to collection of their samples. Runs 1 and 2 were inoculated with fecal extracts from donor A whereas Runs 3 and 4 were inoculated with fecal extracts from donor B and C respectively. Run 5 was inoculated with a defined mixed culture, called Microbial Ecosystem Therapeutic-2 (MET-2), which was a defined community consisting of strains derived from donor A's feces (Table 7-1). Runs 1, 2, 3, and 4 were maintained for 48, 53, 41 and 53 days respectively, while Run 5 was maintained for 27 days. Based on previous studies using this system the community cultures reached steady-state as early as 4 weeks after inoculation[5].

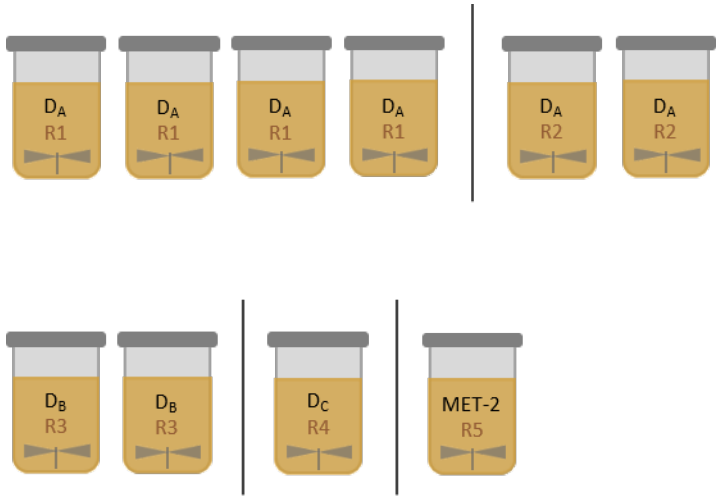

Figure 7-1. Experiment overview. CSTR Bioreactor vessels were inoculated with a seeding culture that were derived from either a human fecal samples, or a defined mixed culture. Fecal samples were obtained from donors A, B and C, as denoted by D_A, D_B, and D_C respectively. A cocktail of defined seeding cultures were used, termed Microbial Ecosystem Therapeutic-2 (MET-2). Bioreactors that were operated in parallel are considered to be part of a single Run (denoted as R1, R2 etc.), where the seeding culture in each bioreactor within a Run are derived from the same inoculant. Samples were collected from bioreactor effluent at various time points during bioreactor operation.

Table 7-1. A list of cultured bacterial isolates from the feces of donor A included in the MET-2 defined community.

Higher taxonomic group	Closest species match, inferred by alignment of 16S rRNA sequence to GreenGenes database	% identity to closest match	Relative abundance[4] (by biomass)
Actinobacteria	*Bifidobacterium longum*	100	1
	Collinsella aerofaciens	100	0.5
	Microbacterium schleiferi	99.34	1.5
	Aldercreutzia equolifaciens	99.76	1
	Micrococcus luteus	97.04	1
Bacteroidetes	*Bacteroides ovatus*	100	0.5
	Parabacteroides merdae	100	1
Firmicutes	*Eubacterium rectale*	100	1
	Blautia luti	98.91	1
	Roseburia hominis	99.04	1
	Roseburia lactaris	95.07	0.5
	Ruminococcus albus	96.96	1
	Eubacterium eligens	96.78	1
	Ruminococcus torques (two different strains)	99.27 100	1 1
	Roseburia intestinalis	100	1
	Eubacterium fissicatena	97.67	1
	Eubacterium ventriosum	97.37	1
	Blautia coccoides	99.85	0.25
	Blautia hydrogenotrophica	100	0.1
	Dorea longicatena	100	0.25
	Dorea formicigenerans	99.49	0.25
	Clostridium ramosum	96.14	0.25
	Eubacterium limosum	99.25	0.75
	Streptococcus thermophilus	100	0.25
	Bacillus simplex	98.7	0.25
	Coprococcus catus	99.19	0.25
	Flavonifractor plautii	96.21	0.25
	Streptococcus mitis	100	0.3
	Phascolarctobacterium sp.	99.85	0.1
Proteobacteria	*Escherichia coli*	100	0.25
	Parasutterella excrementihominis	100	0.5

[4]Relative abundance nits are presented as the equivalent of 1 X 10 µL loopful of bacterial culture scraped from petri dish plates.

7.2.5.2 Bioreactor Operation and Inoculation

Up to four 500-mL Multifors bioreactors (3 Multifors systems, Infors AG, Switzerland), with working volumes of 400 mL were operated as CSTRs. Conditions were set to mimic the distal human gut (37ºC, pH 7, gentle agitation, oxygen-free, and fed with a constant supply of mucin and insoluble starch substrates at a flow rate of ~400 mL/day). Vessels were maintained anaerobic by bubbling nitrogen. Operation and maintenance of the bioreactors were conducted as previously specified[5]. The bioreactors were inoculated with either the defined mixed culture, MET-2, or undefined mixed culture obtained from homogenized fecal samples from human donors. 100 mL of inoculum was added to 300 mL of media to comprise a 10% w/v culture. Cultures were gently agitated pH adjusted to 6.9-7.0. The media was prepared in-house, as previously described[5].

7.2.5.3 Preparation of Fecal Inocula and Defined Mixed Culture

Fecal inocula were prepared as previously described[5], where donors provided fresh fecal samples. The MET-2 consortium was developed by the Allen-Vercoe group of the University of Guelph as a simplified version of the gut microbiota of donor A, and was composed of 33 different strains of bacteria that were isolated from a fecal sample obtained from donor A (Table 7-1). These strains were isolated through plate culture of diluted fecal samples on various media types (deMann Rogosa & Sharpe agar, Brain Heart Infusion agar (Oxoid), Fastidious Anaerobe Agar (Lab90)), with incubation in an anaerobe chamber (Ruskinn) under an atmosphere of N_2:CO_2:H_2 (90:5:5). Isolated colonies were streak-purified and characterized by Sanger sequencing of their 16S rRNA genes with comparison to the Greengenes database (http://greengenes.lbl.gov/cgi-bin/nph-index.cgi). Each inoculum was prepared by scraping biomass from select plate cultures and suspending it in pre-reduced CSTR growth medium (75 mL).

7.2.5.4 Sample Collection and Processing

Bioreactor effluent from Run 1 was pooled over a 10-day period and samples were taken from this pooled material. Samples from CSTR Runs 2-5 were collected on a daily basis; sampling volumes did not exceed 2.5% of the total working volume (so not to impact the chemostatic conditions of the bioreactor). Samples were stored at -80 °C, and thawed prior to analysis in a step wise manner (5 hours in -40 °C, overnight in -20 °C and then on ice). Samples were centrifuged (3800 x g, 10 min) to remove cells, then subjected to ultracentrifugation (14650 x g, 4 °C) to remove media particulates. Finally, samples were filtered using 0.22 µm syringe filters (Acrodisc® Supor membrane, Pall Corporation, UK).

7.2.5.5 NMR Spectra Acquisition and Processing

Samples were diluted to contain 10% (v/v) internal standard in preparation for NMR spectroscopy. The internal standard used was a chemical shape indicator (CSI), a solution of 99.9% D_2O with 5 mM 4,4-dimethyl-4-silapentane-1-sulfonic acid (DSS) and 0.2% (w/v) sodium azide to inhibit bacterial contamination. Sample solutions were transferred to 5-mm glass NMR tubes (NE-UL5-7, New Era Enterprises Inc., Vineland, NJ, USA) and inserted into a Bruker Avance 600.13 MHz spectrometer with a Triple Resonance Probe (TXI 600) for scanning. Samples were stored at 4 °C. Prior to scanning, samples were allowed to warm to room temperature. NMR spectra were acquired using the first increment of a 1D NOESY pulse sequence with t_{mix} of 100 ms, 4 second acquisition time, 1 s relaxation delay, and a spectral width of 12 ppm.

Following acquisition, spectra were imported into Chenomx NMR Suite 7.5 (Chenomx Inc., Edmonton, Canada) for spectral processing and compound identification and quantification. Phase and baseline corrections were carried out manually referring to the internal standard, DSS, and by

assessing clusters at random across the span of the spectra to make the appropriate adjustments. Shim and chemical shape correction was performed automatically by the software based on a manual comparison of ideal and observed DSS peaks. The known concentration of DSS was used to determine the concentration of all other compounds in the spectrum. Sample pH measurements that were obtained during the sample preparation stage were used for spectrum profiling.

7.2.5.6 Metabolite Profiles

Once spectral processing was completed, compounds were identified and quantified by targeted profiling using Chenomx NMR Suite, 7.0-7.7 (Chenomx Inc., AB). The identification process consisted of manually matching the projection of a predicted compound, selected from the software's built-in 600 MHz compound library, to the processed 1D-^{1}H NMR spectral signatures. Criteria for placing a compound on the spectrum included both the fit of the projected signature to spectral clusters and background information on metabolite properties, biofunction, pathways and uses. Such metabolite information was obtained from the Chenomx software database, The Human Metabolome Database (HMDB, www.hmdb.ca) and the Kyoto Encyclopedia of Genes and Genomes (KEGG, www.kegg.jp). Quantification of metabolite concentrations were determined by the area of the projected signal once it was fit to the peak centers during identification [39,41]. As a post hoc evaluation of the metabolite profiles, each compound was assessed based on the number of clusters in the compound's signal, the degree of convolution of the signal with other compounds' signals and the signal to noise ratio. These three characteristics influence the amount of error associated with the identification and quantification process of targeted profiling. Therefore, only compounds that were ranked with high-confidence were included in subsequent analyses of the metabolite profiles.

7.2.5.7 Statistical Analysis

The metabolite profiles were analysed using Principal Component Analysis (PCA) to model the variance between different mixed cultures, and metabolites associated with observed variances. Mean centering ($x - \bar{x}$) and unit variance scaling (x/s, where s is the standard deviation of x concentrations) were applied to the metabolite profiles to easily compare the concentration changes and evaluate all compounds as equally weighted. Metabolite profiles were compared by pair-wise Tukey's Honest Significant Difference (HSD) test where significance was defined by a Bonferroni correction to the alpha value ($p \leq 0.008$). Compounds were identified as a marker when the concentrations in one community's metabolite profile were significantly different from that of all other populations' profiles and where this cross-profile significance only occurred for one community culture. Since the concentrations in the metabolite profiles spanned several orders of magnitude, the compounds within a profile were separated into five bins (20+ mM, 10-20 mM, 1-10 mM, 0.1-1 mM and 0-0.1mM). These bins were defined by a kernel density estimation of compounds' mean concentration, using a bandwidth of 0.031 (Figure A7). The reason for examining the compounds with other compounds of similar magnitude is two-fold. The first is that it allows clear contrast in seeing small increments in concentration changes when visualizing the profiles. The second is that large changes in high-concentration compounds are more likely to be influential than similar percent changes in low-concentration compounds. While this sort of weighting is not desired when analyzing variances and model-building with PCA, it is still important to acknowledge the effect size differences between different compounds when interpreting the changes. All statistical analysis was executed using R, a statistical computing and graphics tool[145]. All graphics were generated by the ggplot2 package[146].

7.2.6 Results

7.2.6.1 Multivariate Analysis

In total, 40 metabolite profiles were acquired from the samples collected. PCA was used to visualize samples given the variation in metabolite profiles. This was done via a score plot (Figure 7-2), where each sample is represented by a point on the graph, and its coordinates on the graph demonstrates the relationship of that sample to others given the covariance matrix of the metabolite profiles. Samples from different donors clustered distinctly from each other, and samples of the same donor clustered together. Even the samples of donor A, which were obtained from two different runs, demonstrated similar metabolite profiles and clustered close together. Given that principal component (PC) 1 and 2 cumulatively account for 64.8% of the total variation explained, while PC 3 accounts for only 11.3% of the total variation explained, it can be concluded that donor variability, rather than run-to-run variability, is the main contributor to the amount of variation observed in the metabolite profiles.

Additionally, it became evident that the metabolite profiles derived from human fecal communities were more similar to each other than the profiles of defined communities, despite the latter being designed to resemble the consortia of one of the human donations. Profiles of human feces-derived cultures differed from profiles of the defined culture along PC1 on the x-axis, which accounted for 39.7% of the total variation. The compounds that drove the relationships depicted in the score was extracted from the corresponding loading plot, where the coordinate location of the compounds describes how those compounds contribute to those sample relationships (Figure A8B). The similarities and differences in cultures were also apparent through a global view of the metabolite profiles comparing mean compound concentrations across different donors (Figure 7-3). The compounds most influential in differentiating feces-derived communities and defined mixed cultures

are listed in Table 7-2. Carboxylic acids and derivatives, fatty acids and conjugates, and amino acids and derivatives were the most common types of compounds that differentiated the two groupings.

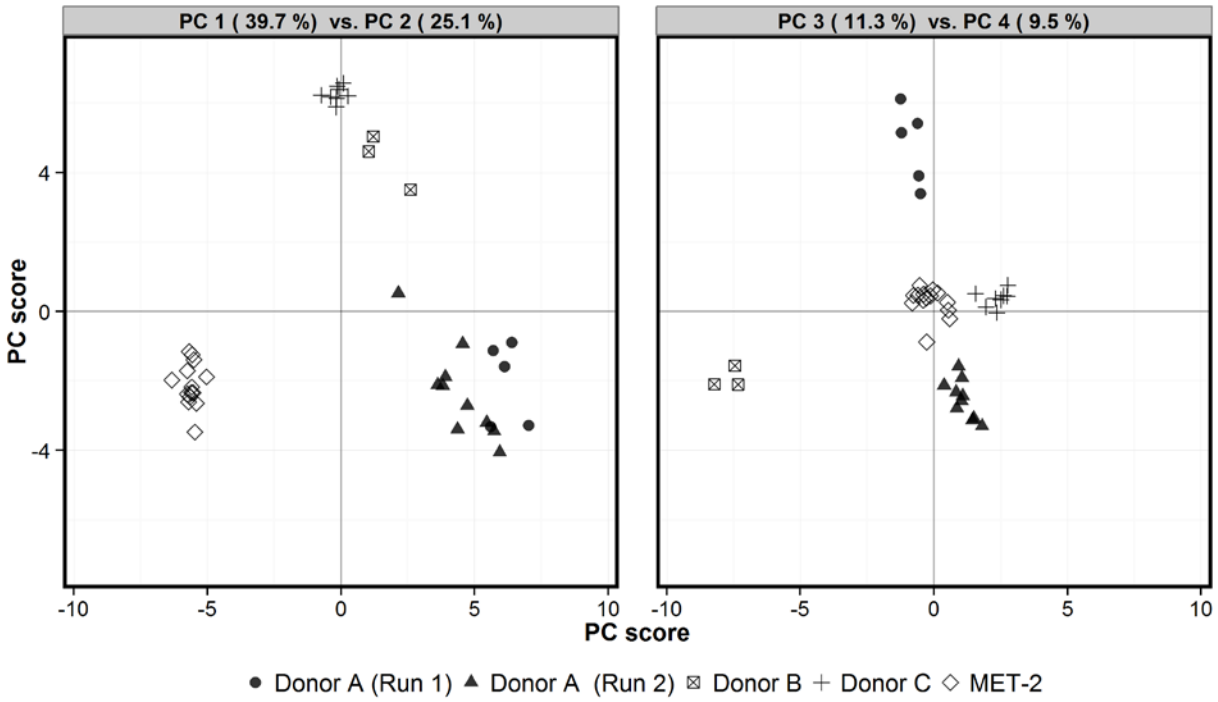

Figure 7-2. Principal component analysis score of metabolite concentration profiles obtained from samples of bioreactor effluent. Metabolite concentrations have been mean centered and unit variance scaled. Seeding culture sources are represented by symbols, which are coloured by runs, where a run includes several bioreactor vessels seeded with the same inoculant and operating in parallel. Principal component (PC) 1 and PC 3 are represented along the x axis, while PC 2 and PC 4 are represented along the y-axis. PC 1 and PC 2 account for 39.7% and 25.1% of the total variation, respectively. PC 3 and PC 4 account for 11.3% and 9.5% of the total variation respectively.

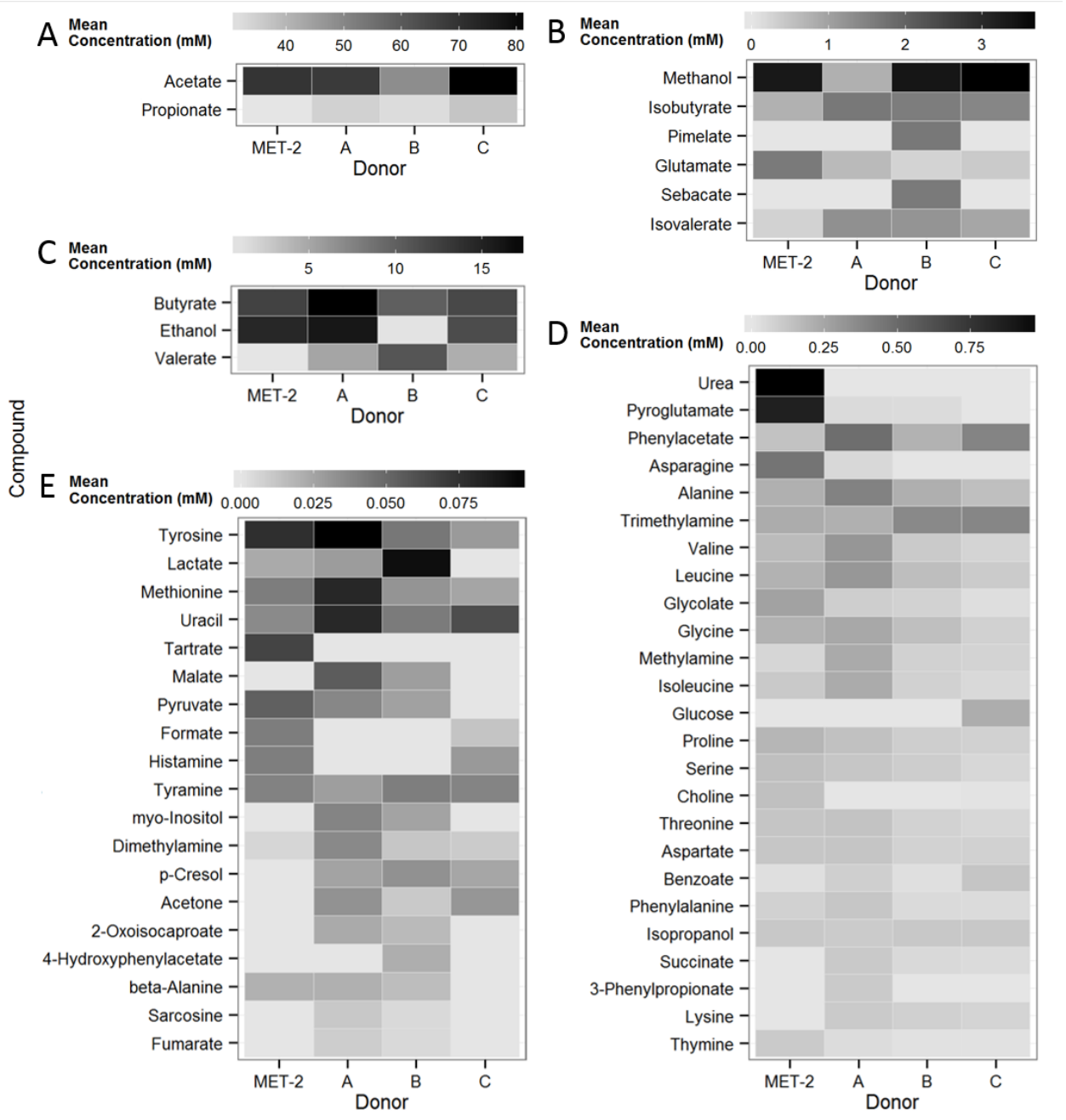

Figure 7-3. Metabolite profile comparison different donors. The mean compound concentrations (mM) of samples from the untreated conditions are presented for each donor. The profiles are split into five bins, as defined by kernel density plot of mean concentrations (Supplementary figure). (A) Bin range: mean ≥ 20 mM. (B) Bin range: 1 ≤ mean < 10 mM. (C) Bin range: 10 ≤ mean < 20 mM. (D) Bin range: 0.1 ≤ mean < 1 mM. (E) Bin range: 0 ≤ mean < 0.1 mM.

Table 7-2. Compounds that differentiated donor-derived cultures and defined cultures as tested by the Mann-Whitney test at the 95% confidence level.

	Compound[5]	P-value	Compound Class
Bin 1	Propionate	2.76E-06	Carboxylic Acids and Derivatives
Bin 2	Valerate	8.62E-11	Fatty Acids and Conjugates
Bin 3	Isobutyrate	8.62E-11	Carboxylic Acids and Derivatives
	Isovalerate	8.62E-11	Fatty Acids and Conjugates
	Glutamate	2.65E-07	Amino Acids and Derivatives
Bin 4	Phenylacetate	8.62E-11	Phenylacetic Acid Derivatives
	Methylamine	1.72E-10	Alkylamines
	Urea	1.52E-09	Ureas
	Asparagine	3.77E-08	Amino Acids and Derivatives
	Choline	6.70E-08	Alcohols and Polyols
	Succinate	1.44E-07	Carboxylic Acids and Derivatives
	Pyroglutamate	2.38E-07	Pyrrolidines
	Thymine	2.61E-07	Diazines
	Glycolate	2.65E-07	Hydroxy Acids and Derivatives
	Benzoate	3.78E-07	Benzoic Acid and Derivatives
	Lysine	1.10E-05	Amino Acids and Derivatives
	Proline	4.17E-05	Amino Acids and Derivatives
	Serine	0.000155	Amino Acids and Derivatives
	3-Phenylpropionate	0.000655	Cinnamic Acid Derivatives
	Trimethylamine	0.017892	Alkylamines
	Glucose	0.024295	Monosaccharides
Bin 5	Tartrate	1.52E-09	Carboxylic Acids and Derivatives
	Formate	6.73E-08	Carboxylic Acids and Derivatives
	Histamine	1.10E-07	Azoles
	p-Cresol	1.44E-07	Phenols and Derivatives
	Dimethylamine	3.07E-07	Alkylamines
	Acetone	3.70E-07	Carbonyl Compounds
	2-Oxoisocaproate	9.72E-05	Fatty Acids and Conjugates
	Malate	9.73E-05	Carboxylic Acids and Derivatives
	Fumarate	9.73E-05	Fatty Acids and Conjugates
	myo-Inositol	9.73E-05	Cyclic Alcohols and Derivatives
	Sarcosine	9.73E-05	Amino Acids and Derivatives

[5]Compounds are listed in order of increasing p-value within each compound bin. Bin 1: $\geq$ 20 mM; Bin 2: $\geq$ 10 mM, < 20 mM; Bin 3: $\geq$ 1 mM, < 10 mM; Bin 4: $\geq$ 0.1 mM, < 1 mM; Bin 5: > 0 mM.

	Pyruvate	0.000227	Keto-Acids and Derivatives
	Tyramine	0.017207	Phenethylamines
	Uracil	0.022443	Diazines
	beta-Alanine	0.037663	Amino Acids and Derivatives

7.2.6.2 Metabolite Profiles

Amino acids and fatty acids were the two classes of compounds most commonly profiled in each of the cultures and the abundance of these compounds was a trait common to metabolite profiles of every bacterial community investigated. On average, feces-derived cultures had more compounds in their metabolite profiles compared to that of the defined culture; donor A, B and C had 47, 48 and 40 different compounds respectively and MET-2 had 42 different compounds. When the frequency of each compound class in the metabolite profiles were weighted to the respective concentrations assigned, carboxylic acids and derivatives, fatty acids and conjugates and alcohols and polyols occupied the largest proportions of each culture's profiles (Table 7-3)

Table 7-3. Metabolite profile diversity of each bacterial culture. Proportion of each compound class (%) weighted to the compounds' concentration.

Compound Class	**Number of Compounds**	**Bacterial Culture**			
		A	B	C	MET-2
Carboxylic Acids and Derivatives	9	69.59801	74.72572	77.31247	72.48956
Fatty Acids and Conjugates	5	15.95946	19.2173	11.08216	9.650773
Alcohols and Polyols	4	11.324	3.840536	9.901108	13.1407
Amino Acids and Derivatives	18	2.067571	1.280227	0.794359	2.670579
Phenylacetic Acid Derivatives	1	0.323521	0.170689	0.250661	0.09505
Alkylamines	3	0.316235	0.392174	0.295705	0.20722
Hydroxy Acids and Derivatives	2	0.077436	0.142395	0.016511	0.206701
Diazines	2	0.074816	0.046046	0.04838	0.099248
Cinnamic Acid Derivatives	1	0.069176	0	0	0
Benzoic Acid and Derivatives	1	0.055678	0.018187	0.077969	0.016401
Pyrrolidines	1	0.027272	0.031161	0	0.614319
Cyclic Alcohols and Derivatives	1	0.025528	0.022009	0	0
Keto-Acids and Derivatives	1	0.024497	0.022676	0	0.038515
Carbonyl Compounds	1	0.021668	0.00918	0.019653	0
Phenethylamines	1	0.018246	0.035099	0.024367	0.027895
Phenols and Derivatives	2	0.016894	0.046596	0.015592	0
Azoles	1	0	0	0.018856	0.029123
Monosaccharides	1	0	0	0.142208	0
Ureas	1	0	0	0	0.713916

In fact, the proportional distribution of all the compound classes were very similar between all four types of cultures. One of the few discrepancies between the cultures was that donor B-derived culture profiles contained 3.84% alcohols and polyols while all other cultures had ~10% or more of this class of compounds. While characteristics such as compound class composition and number of metabolites profiled could be generalized, each population of enteric bacteria still produced unique metabolite profiles that could be described based on several identifying features.

Each of the communities' metabolite profiles had unique features where certain metabolites could be identified as a marker for that particular bacteria population. 3-Phenylpropionate was unique to the metabolite profiles of donor A-derived cultures as it was not found in any other cultures' metabolite profiles. Donor A-derived samples also had elevated levels of leucine, valine, isoleucine, methylamine, phenylalanine and methionine ($p \leq 0.008$). Mean methanol levels were much lower, while mean succinate, and dimethylamine levels were higher in donor A than all the other cultures, however this was not statistically significant due to relatively large standard deviations.

Donor B culture had three markers that were unique to its metabolite profile. Pimelate and sebacate, both dicarboxylic acids, were found at 1.63 mM and 1.60 mM mean concentrations, respectively, but were absent in the profiles of communities other than Donor B. 4-Hydroxyphenylacetate, an aromatic acid, was found only in donor B culture's profile, though at very low concentrations (mean 0.023 mM). Other identifying features of donor B community's metabolite profile was decreased levels of acetate and ethanol. Donor B cultures had an average concentration of 48.6 mM of acetate and 0.91 mM of ethanol, whereas other communities had significantly greater levels of acetate and ethanol, in the range of 67.62-81.45 mM and 11.5-16.05 mM, respectively. The donor B community also had the highest levels of valerate and lactate of all the communities. The mean concentration of valerate in the metabolite profiles of donor B communities was 10.8 mM,

while the mean of donor A, C and MET-2 communities were 5.02 mM, 4.41 mM and 0.80 mM, respectively. Similarly, the mean lactate concentration in donor B was 0.093 mM, while the mean lactate concentration in donor A and MET-2 communities were 0.028 mM and 0.023 mM respectively.

Donor C was the only community that yielded metabolite profiles that did not contain any lactate. Similarly, donor C culture profiles did not contain any beta-alanine nor pyruvate, while all other cultures' metabolite profiles had mean concentrations within the range of 0.015-0.026 mM and 0.026-0.054 mM respectively. On the other hand, donor C culture's profile was the only one that contained glucose (mean 0.23 mM).

The MET-2 community was the only culture that produced a metabolite profile that contained tartrate and urea. Furthermore, MET-2 culture metabolite profiles contained significantly elevated levels of glutamate, pyroglutamate, asparagine, glycolate, choline, thymine and formate compared to donor A, B and C metabolite profiles ($p \leq 0.008$). Figure 7-4 provides further detail on these identifying markers.

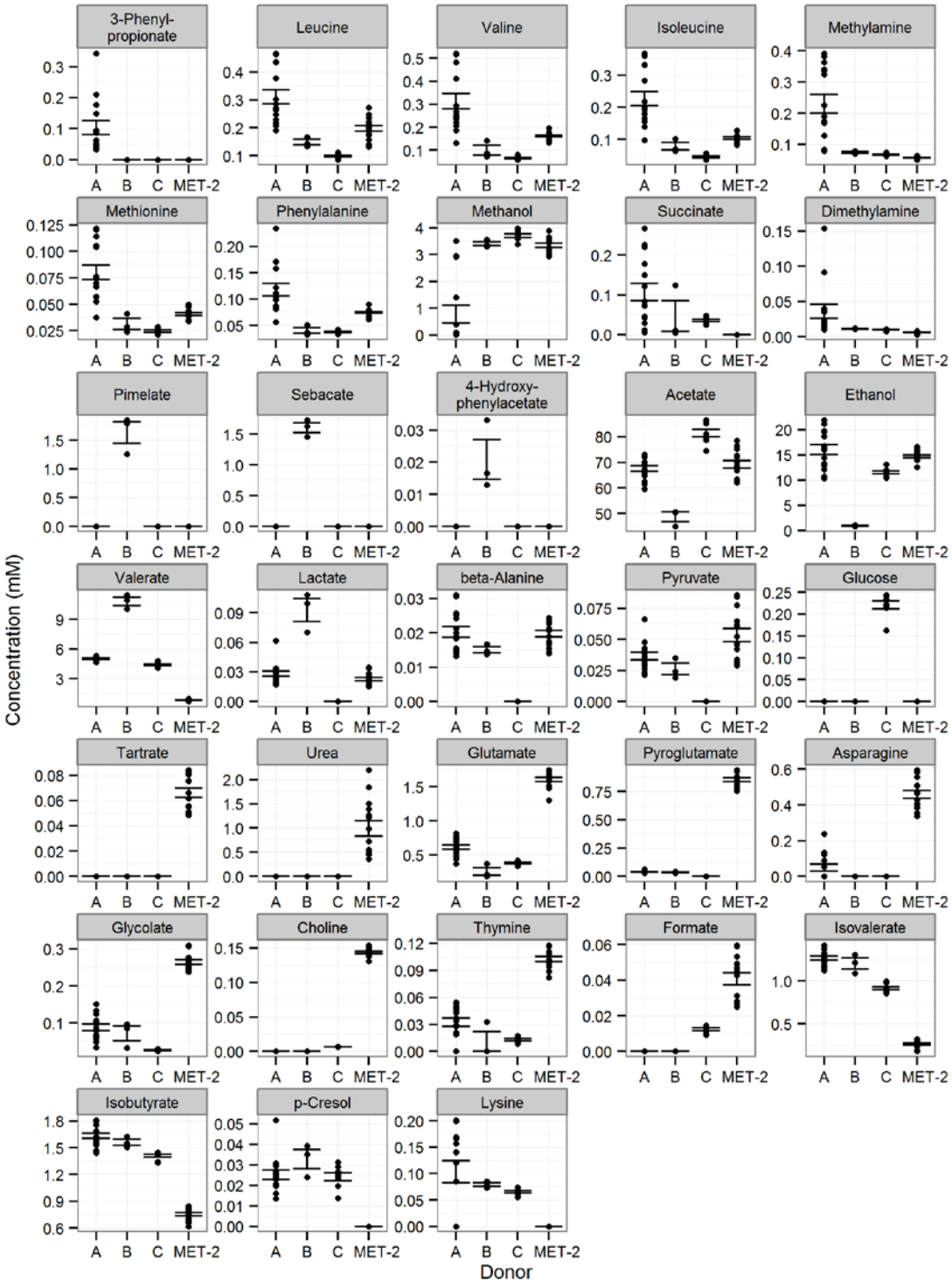

Figure 7-4. Metabolites as identifying markers for metabolite profiles.

7.2.7 Discussion

7.2.7.1 Feces-Derived vs Defined Community Cultures

Metabolite profiles of feces-derived samples were more similar to each other than MET-2 (defined community) samples. Given that the defined community was a subset of isolates derived from a fecal community and which therefore displayed less alpha diversity than its parent community[147], differences in the metabolite profiles between defined and fecal communities were expected. According to multivariate analysis, where compounds are weighted equally, the compounds that were found at greater concentrations in MET-2 compared to fecal cultures were mostly low-concentration compounds (mean < 1 mM) (Figure A8B). The low concentrations of these compounds indicate that these metabolites could be intermediates in major pathways. While these compounds were statistically significant, their impact on the bacterial culture as a whole could be regarded as minimal. Glutamate was the only compound higher in MET-2 culture profiles (mean equal to 1.61 mM) compared to feces-derived culture profiles (means ranged from 0.26 to 0.62 mM). Glutamate is utilized by various species in Groups I through IV of the Clostridia class, forming acetate, butyrate, carbon dioxide, ammonia and hydrogen as products[148,149]. While acetate and butyrate were observed in abundance, the latter three compounds were not measured in the metabolite profiles due to their incompatibility with ^{1}H NMR spectroscopy. On the other hand, glutamate synthesis is associated with osmoregulation[150,151] and ammonia assimilation[152] in anaerobic bacteria. Therefore, the difference in glutamate levels between the feces-derived and defined cultures could be indicative that the communities have differences in their propensity to perform these glutamate-associated biofunctions. Since the biofunctions associated by glutamate synthesis involve fluctuation of intracellular glutamate, those pathways are unlikely to be reflected in the metabolite profiles analyzed here, which represent the exometabolome only. Even though acetate and butyrate were end-products of glutamate

breakdown, they were not observed to be at significantly lower levels in MET-2. This is likely due to the fact that acetate and butyrate are also by-products of other fermentation pathways, such as breakdown of other amino acids [149]. Therefore, the concentration of acetate and butyrate observed is the cumulative result of the entire community's metabolic reactions.

Comparison between the two types of cultures also revealed that high-concentration compounds (mean $\geq$ 1 mM) were found at decreased levels in defined cultures compared to fecal cultures. Since high-concentration compounds were the major components of the metabolite profiles, disparities in those compounds between the two groups are both significant and important. Propionate, valerate, and isovalerate have mean values of 34.7 mM, 6.74 mM, and 1.13 mM respectively in fecal cultures and 30.7 mM, 0.802 mM, and 0.236 mM respectively in the defined culture. Propionate, valerate, isobutyrate, and isovalerate are short chain and branched chain fatty acids (SCFA and BCFA respectively) previously reported to be produced by anaerobic bacteria of the distal GI tract[74,153,154]. Propionate fermentation occurs via the acrylate pathway in members of the *Clostridium* genus or via the succinate-propionate pathway in other bacteria such as *Bacteroides fragilis*[155,156]. Since these SCFA and BCFAs were not found in the metabolite profile of the CSTR growth medium, all the propionate, valerate, isobutyrate and isovalerate detected was produced by bacterial metabolic activities.

7.2.7.2 Metabolite Profiles of Bacterial Communities

The metabolite profiles can be broken down into metabolite classes as classified by HMDB (Table 3). The profiles can be viewed as a composition of chemical taxonomy classes proportional to the concentration of each compound. The proportion of compound classes that make up the metabolite profiles was consistent across all the bacterial cultures analyzed, where carboxylic acids and derivatives, fatty acids and conjugates and alcohols and polyols were present in the highest

concentrations, cumulatively accounting for at least 95% of the metabolite profiles. Variability in the metabolite composition occurred in the remaining 2-5%% of the profiles. For example, amino acids occupied ~2% of the metabolite profiles in donor A-, and B-derived community and the defined community, but in the donor C-derived community made up only 0.79% of the metabolites. Less than 4% of the metabolites from donor B-derived cultures were alcohols and polyols, but made up ~10% or more of the metabolites in the profiles from other cultures. Despite these small variations, the consistency of the metabolite profiles' structure could be an indication that the intestinal communities share a functional steady state. Therefore, the metabolic interactions tend towards satisfying these functional roles, even when the microbial composition of the community are different.

While the general structure of the metabolite profiles based on the compounds' chemical taxonomic classes could be a characteristic of communities of intestinal bacteria, it does not preclude the fact that there are important metabolic differences between various intestinal microbial ecosystems that are reflected in their metabolite profiles. Such are distinctive and could be used as identifying markers.

7.2.7.3 Donor A

The microbial ecosystem derived from donor A was elevated in compounds involved in the valine-leucine-isoleucine metabolism pathway and the presence of 3-phenylpropionate (absent in all other fecal community samples, including MET-2) suggested an active phenylalanine metabolism pathway[157,158]. 3-Phenylpropionate is known to be produced by 13 different species in the Clostridiales order, including *C. sporogenes, C. caproicum* and *C. botulinum* (*Clostridiaceae* family), and *C. bifermentans* and *C. sordellii*, (*Peptostreptococcaceae* family)[133,159], none of which are components of the MET-2 community (Table 1). Alternatively, 3-phenylpropionic acid has been demonstrated to be catabolized by *Escherichia coli* to produce acetaldehyde and pyruvate[157]. Given

that MET-2 was designed to mimic the donor A community, and both communities contained *E. coli*, it can be assumed that both donor A and MET-2 communities would have similar, if any, 3-phenylpropionate catabolism capabilities. Therefore, the absence of 3-phenylpropionate in all culture profiles except that of donor A's was likely due to increased phenylalanine dissimilation capabilities, rather than decreased usage of 3-phenylpropionate.

7.2.7.4 Donor B

The metabolite profile of donor B could be distinguished by the presence of pimelate, sebacate and 4-hydroxyphenylacetate. Derivatives of pimelate have been shown to be part of the lysine biosynthesis pathway in organisms such as *E. coli and Bacillus subtilis* and the biotin biosynthesis pathway in *B. spaericus*[160,161]. Bacteria that are able to oxidize long-chain dicarboxylic acid compounds, such as sebacate, are also capable of metabolizing a wide range of dibasic acids, like pimelate, suberate and azelate[162]. 4-Hydroxyphenylacetate is an aromatic acid that has been demonstrated to be part of the phenylalanine-tyrosine pathway of *Klebsiella pneumoniae* and the denitrifying pathway of *Pseudomonas* spp.[163,164]. Acetate and ethanol are produced by fermentation activities of enteric bacteria, and also act as the main substrate for sulfur-reducing enteric bacteria[154,165]. Given these observations, the donor B-derived community appeared to have greater dicarboxylic metabolism capabilities than the other communities studied, which in turn could be related to either increased levels of fermentation activity or decreased presence of sulfate-reducing bacteria. Valerate and lactate were identified as elevated in donor B-derived communities, and both are associated with fermentation; valerate is a SCFA produced from metabolism of carbohydrates and amino acids[166] and lactate is a hydroxy acid that is produced alongside other organic acids during fermentation[154,165].

7.2.7.5 Donor C

Interestingly, neither lactate nor pyruvate were detected in the donor C-derived community metabolite profile, even though these compounds were found in all other communities studied. These are common compounds within microbial communities, however, and we speculate that their concentrations were below the level of detection using our methods, rather than being completely absent. The donor C community metabolite profile was also the only one that still contained glucose. Since complex carbohydrates were the carbon source available in the feed medium, presence of glucose in the effluent samples was indicative of successful breakdown of these polysaccharides but reduced propensity to take up simple sugars. Thus, the absence of lactate and pyruvate as well as the presence of residual glucose all indicated that the donor C microbial community had not metabolized all of the simple sugar. The lack of beta-alanine was also unique to the donor C community. Several organisms, including *C. sporogenes*, follow a reductive pathway in degrading pyrimidines to produce beta-alanine[167], and in some microorganisms beta-alanine can be subsequently broken down to be used as a nitrogen source[168,169]. As such, the absence of beta-alanine in the donor C profile could be attributed to either a non-reductive pathway of degrading pyrimidines or the inability to utilize beta-alanine.

7.2.7.6 MET-2

MET-2 was the only bacterial population that had tartrate in its metabolite profiles. Tartrate is a dicarboxylic acid known to contribute to fecal pH[170]. Certain anaerobic bacteria, including some species found in feces, are able to breakdown tartrate to produce SCFAs, ethanol, lactate, carbon dioxide and hydrogen[171–174]. The production of tartrate has been studied extensively in *Gluconobacter suboxydans*, a strict aerobe, for industrial purposes[175,176]. While *G. suboxydans* is not a gut microbe, its tartrate biosynthesis pathway (from glucose, with glycolate as a by-product) could be similar to

those in the MET-2 community. Indeed, glycolate was found in MET-2 profiles at levels an order of magnitude greater than in all other cultures. Based on the evidence that the presence of tartrate was unique to the MET-2 community, and that glycolate was elevated in the MET-2 culture, it appears that the MET-2 community has greater tartrate producing capabilities, rather than a diminished level of tartrate utilization.

Urea was also a unique characteristic of the MET-2 culture. Urea is a waste product from both purine and amino acid degradation, and is usually degraded by bacterial urease to generate free ammonia[177,178]. Amino acid degradation products were detected in all studied cultures, but urea was only detected in MET-2. We speculate that because MET-2 has low diversity compared to fecal communities, the MET-2 culture lacked the ability to perform urea hydrolysis, rather than had an enhanced ability to break down amino acids.

MET-2 cultures also had higher concentrations of glutamate, pyroglutamate, asparagine, choline, thymine, glycolate and formate. Glutamate and pyroglutamate both participate in glutamate metabolism, such as ammonia assimilation. Asparagine is known to be associated with generation of ammonia via an asparagine-aspartate deamidation pathway[179,180]. Therefore, an abundance of asparagine could be an indication of decreased activity along the asparagine degradation pathway. Another amino acid that was elevated in MET-2 cultures compared to other communities was choline. In anaerobic bacteria, choline utilization involves its enzymatic cleavage to produce trimethylamine, which can be subsequently used by methanogenic bacteria[181–183]. Significantly decreased levels of trimethylamine in the MET-2 samples compared to the donor B and C derived communities were most likely because there was a reduced capacity in the MET-2 community to cleave choline. Interestingly, the donor A community, from which MET-2 was derived, also had low levels of trimethylamine, indicating that this characteristic could be preserved in a derivative community.

Thymine is a pyrimidine that is utilized as a nitrogen source via either reductive or oxidative pathways[184,185]. Given that the MET-2 community exhibited pyrimidine levels over 3 times greater than that of the donor A samples, and 10 times greater than the donor B and C samples, we conclude that MET-2 was not able to catabolize thymine to the same extent as other community cultures. Glycolate and formate were also found to be significantly higher in MET-2 than other cultures. Glycolate, as well as formate, is involved in energy metabolism during fermentation of carbohydrates and degradation of amino acids[149,186]. Although, as described above, glycolate presence may be attributed to tartrate production. Even though both formate and glycolate were observed at a concentration less than 1 mM in MET-2 samples, this level was still relatively high compared to the profiles of donor A, B or C-derived communities. Subtle differences in metabolically linked compounds, such as glycolate and formate could be representative of differences in the metabolic interactions occurring within each culture.

7.2.8 Conclusion

Communities of colonic bacteria propagated *in vitro* were described by their metabolite profiles, which were built using 1D-^{1}H NMR spectroscopy. Overall, the metabolite profiles were consistent in the proportions of compound classes, where carboxylic acids (69.6-77.3%), and fatty acids (9.7-19.2%) were the most abundant types of metabolites. The two types of cultures (defined or feces-derived) were seen to have different sugar metabolism capabilities, based on the levels of SCFAs (propionate and valerate) and BCFAs (isovalerate and isobutyrate). The simpler defined community, MET-2, was less proficient at carbohydrate metabolism compared to its feces-derived community counterparts. In general, more compounds were found in the feces-derived cultures, though many that were unique to the feces-derived cultures were found at very low concentrations, less than 0.1 mM.

Ultimately, the features within each metabolic profile not only identified the community culture, but also allowed inferences on their respective metabolic capabilities to be made.

7.2.9 Acknowledgements

This work was supported in part by the generous contribution of Chenomx Inc. in the form of Chenomx NMR Suite 7.0-7.7 software. Acknowledgements are also extended to NSERC Discovery, NSERC Strategic Network (MabNet), and a University of Waterloo's Chronic Disease Prevention Initiative Seed Grant (PROPEL Center for Population Health Impact) Grants to MGA, and an NSERC Canada Graduate Scholarship to SS. Microbiology and CSTR culture work was supported by NSERC Discovery and OMAF/University of Guelph partnership grants to EAV, a CIHR Doctoral Research Award to JAKM, and an Ontario Graduate Scholarship to KS. The authors also thank Michelle Daigneault for her technical expertise in microbial culture and Anita Yen for her artistic contribution to the abstract figure.

7.3 Analysis of Perturbations to Human Fecal Microbiota Propagated *In Vitro*

7.3.1 Experiment Objective

Given that the metabolite profiles were demonstrated to reveal insight into the metabolic functions occurring in *in vitro* microbial communities, further experiments were conducted to assess the robustness of the communities' functional roles. Several different perturbations were administered to feces-derived and defined communities after achieving steady state in the CSTRs. The objective of this study was to characterize the effects of different types of perturbations on bacterial communities based on their metabolite profiles. A wide range of perturbing agents was used to demonstrate the breadth of investigations to which this approach could be used.

7.3.2 Justification of Treatment Condition

7.3.2.1 Clindamycin

Clindamycin is a broad spectrum antibiotic that is active against anaerobic bacteria, such as those found within the gastrointestinal system[187]. The antibiotic inhibits protein synthesis within the cell by binding to the 50S ribosomal subunits and preventing aminoacyl sRNA from binding to the messenger ribosome complex[188]. This mechanism of action is effective against most anaerobes as well as most aerobic gram-positive bacteria (Table A6), though its ineffectiveness against *Clostridium difficile* cautions clinicians to reserve the use of clindamycin for serious infections[189,190]. Due to its potency against gram-negative anaerobic bacteria, clindamycin is an appropriate antibiotic to target microflora in the oxygen depleted, distal parts of the GI tract[190].

7.3.2.2 Vancomycin

Vancomycin is a broad spectrum antibiotic usually reserved for treating serious infections caused by β-lactam-resistant pathogens[191]. Like β-lactam antibiotics, vancomycin targets the mechanisms involved in biosynthesis of bacterial cell walls. While β-lactam antibiotics prevent the crosslinking of peptides to the peptidoglycans of the cell wall, vancomycin operates by specifically targeting the terminal peptides on the peptidoglycans required for transpeptidation reactions in forming the bacterial cell wall[192,193].

7.3.2.3 Norepinephrine

Norepinephrine is a catecholamine that functions as a neurotransmitter or a hormone, depending on the area of the body in which the compound is produced. The production of norepinephrine is under control of the sympathetic nervous system, and activates the fight-or-flight response to influences heart rate and other physiological changes that cue the body for rapid metabolic change and physical movement[194]. As a hormone, norepinephrine travels through the blood to reach their target tissue to reinforce the sympathetic responses[195]. It has been documented that a substantial amount of the body's total norepinephrine production and turnover is performed by mesentery organs, which includes the gastrointestinal tract, spleen and pancreas[196]. Since exposure to stress has been shown to induce a sustained increase in catecholamine levels, the effects of psychological stress on the gut microbiota have been probed regarding the integrity of the microflora and resistance to colonization of pathogenic organisms[197–200]. Specifically, studies in the literature suggest that norepinephrine increases *in vitro* growth of pathogenic gram negative bacteria found in the GI tract: *Yersinia enterocolitica*, *Escherichia coli*, and *Pseudomonas aeruginosa*[201]. It is of interest to investigate if the documented effects of norepinephrine on pure cultures manifest in the same way in community cultures and is able to influence the overall structure of the bioreactor microflora.

7.3.2.4 Defined Bacterial Community

A defined community of bacteria was developed by the Allen-Vercoe group of the University of Guelph as a simplified version of the gut microbiota. Referred to as MET-1, the mixed culture is composed of 33 different strains of bacteria that were isolated from a fecal sample obtained Donor D (Table A1). Using a subset of defined community to perturb the CSTR culture system allows one to monitor the test culture's response to competition.

7.3.3 Materials and Methods

7.3.3.1 Overview of Bioreactor Setup

This study was performed over 6 different bioreactor experimental 'runs', where any given run had one or more bioreactor vessels operating. All vessels within a run propagated cultures from the same seeding source; that is, the inoculant for all vessels in each run was derived from the same inoculum. Conversely, different bioreactor runs used different inoculants. The inocula for Runs 1 and 2 were collected Donor A whereas the inocula for Run 3 was collected from Donor B. Run 5 was inoculated with a defined mixed culture, MET-2, modeled on the fecal community of Donor A (Figure 7-5). Runs 1, 2, and 3 were maintained for 48, 53, and 41 days respectively, while, Runs 5 was maintained for 27 days. Based on previous studies using this system the community cultures reached steady-state by ~5 weeks after inoculation[5].

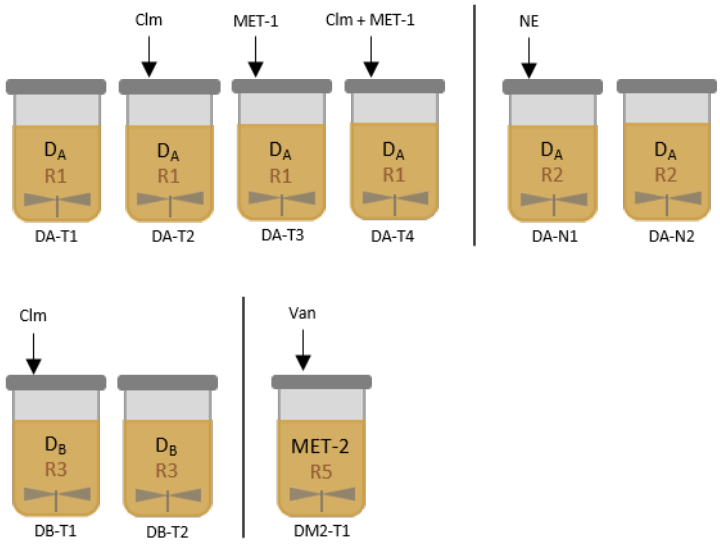

Figure 7-5. Experiment overview. The community culture grown in each bioreactor as labelled as D_A=Donor A, D_B=Donor B, D_C=Donor C, MET 1=Microbial Ecosystem Therapeutic-1, MET-2=Microbial Ecosystem Therapeutic-2, Clm=clindamycin, NE=norepinephrine, Van=vancomycin.

7.3.3.2 Bioreactor Perturbations

One of several perturbing agents were used to treat the mixed cultures by delivering the agent directly to the bioreactor vessel. In Run 2, the Donor-A derived culture (DA-N1) was administered norepinephrine (L-(-)-norepinephrine (+)-bitartrate salt monohydrate, Sigma Aldrich) to a final concentration of 100 mM every 12 hours starting from day 48 post-inoculation until day 52.5 post-inoculation. Similarly, an equal volume of sterile water was added to the sister vessel, DA-N2, as a control. In Run 1, clindamycin (clindamycin 2-phosphate, Sigma Aldrich) was added the Donor A-derived culture (DA-T2) every 12 hours from day 36 to day 40 post-inoculation. The final clindamycin concentration reached at each dosage was 12.6 µg/mL on day 36, 16.6 µg/mL on day 37, 22.4 µg/mL on day 38, 124.7 µg/mL on day 39, and 203.8 µg/mL on day 40. At the same time, the culture DA-T1 was administered dosage-equivalent volumes of sterile water to be assessed as the control vessel. The culture DA-T4 was administered clindamycin from day 36 to day 40 post-inoculation at the same dosages as DA-T2. Following clindamycin treatment, MET-1 was delivered to DA-T4 on day 42 post-inoculation. The MET-1 treatment was prepared as described in Chapter 3, and administered directly into the bioreactor vessel, being careful not to displace the resident bioreactor culture during the treatment delivery process. Concurrently, the DA-T3 community was administered clindamycin-dosage-equivalent volumes of sterile water from day 36-40 and on day 42, was the same MET-1 treatment as DA-T4 was delivered. In Run 3, the culture DB-T1 was administered clindamycin every 12 hours from day 36-40 at the same dosages as those given to DA-T2. As the control vessel, DB-T2 was administered dosage-equivalent volumes every 12 hours from day 36-40. In run 5, the MET-2 culture was treated with vancomycin to a final concentration of 0.5 mg/mL on day 26 post-inoculation.

7.3.3.3 Sample collection and processing

The bioreactor Runs 1, 2, and 3 were studied based on samples from the bioreactor effluent on a daily basis, except for samples from Run 1, where samples were collected from the effluent pooled over a period of 10 days. Runs 4, and 5 were sampled directly from the bioreactor, where sample volumes did not exceed 2.5% of the total working volume so not to impact the chemostatic conditions of the bioreactor. Samples were centrifuged (5000 rpm, 10 min) to remove cells, then subjected to ultracentrifugation (37 000 rpm, 4°C) to pellet media particulates. Finally, samples were filtered using 0.22 µm syringe filters (Acrodisc® Supor membrane, Pall Corporation, UK) to remove bacterial cells. Samples were stored in 4°C until time of NMR analysis.

7.3.3.4 Statistical analysis

The metabolite profiles were analyzed using principal component analysis (PCA) to model the variance between different mixed cultures, and metabolites associated with observed variances. Mean centering (x-$x - \bar{x}$) and unit variance scaling (x/s, where s is the standard deviation of x concentrations) were applied to the metabolite profiles to easily compare the concentration changes and evaluate all compounds as equally weighted. Changes in specific compounds due to perturbations were assessed using the Mann-Whitney U test, at a 95% confidence level. Since the concentrations in the metabolite profiles spanned several orders of magnitude, the compounds within a profile were separated into five bins (20+ mM, 10-20 mM, 1-10 mM, 0.1-1 mM and 0-0.1mM). These bins were defined by a kernel density estimation of mean concentration of compounds in untreated samples, using a bandwidth of 0.03089 (Figure A7). The reason for examining the compounds with other compounds of similar magnitude is two-fold. The first is that it allows clear contrast in seeing small increments in concentration changes when visualizing the profiles. The second is that large changes in high-concentration compounds are more likely to be influential than similar percent changes in low-

concentration compounds. While this sort of weighting is not desired when analyzing variances and model-building with PCA, it is still important to acknowledge the effects size differences between different compounds when interpreting the changes. All statistical analysis was executed using R, a statistical computing and graphics tool.

7.3.4 Results

7.3.4.1 Antibiotic treatment

Administration of clindamycin and vancomycin to different community cultures produced several changes in the metabolite profiles. Two different fecal communities were treated with clindamycin, one derived from donor A, and one from donor B. The effect of clindamycin treatment was a substantial change in many of the metabolites (Figure 7-6). In the high concentration compounds (mean $\geq$1 mM), which consisted of mostly short chain fatty acids (SCFA), long chain fatty acids (LCFA), branched chain fatty acids (BCFA) and alcohols, there were few drastic changes observed with clindamycin treatment. Most of the fatty acids decreased by a moderate amount, in the 20-50% range, except for acetate, which increased by 17% in the clindamycin-treated community from donor A. Ethanol and glutamate were the only compounds that increased by more than 100% after clindamycin treatment, and both increases were observed in the donor B community. In the Donor A community, the level of ethanol did not change significantly and glutamate increased by 63%. Of the all compounds profiled at lower concentrations (<1 mM), amino acids and some organic acids were affected the most, experiencing increases of greater than 100%. Succinate, a SCFA, increased by more than 100% after clindamycin treatment. Acetone, a carbonyl, and 2-oxoisocaproate, a fatty acid conjugate, were also low-concentration compounds that experienced large increases. Few low-concentration compounds decreased in concentration with clindamycin treatment, but the ones that did only did so moderately. These included trimethylamine, methylamine, benzoate and

dimethylamine. Some compounds, such as threonine, aspartate, serine, lysine, sarcosine and fumarate, saw significant increases in treatment of Donor B community, but no significant changes in treatment of Donor A community.

The effects of vancomycin were examined by administering the antibiotic to a defined community, MET-2. The overall effects of this treatment were low to moderate decreases in high-concentration compounds: acetate (-11%), propionate (-40%), butyrate (-37%), and isovalerate (-43%). Additionally, there were large increases in several low-concentration compounds: alanine, valine, leucine, glycine, isoleucine, proline, phenylalanine, tyrosine, lactate, methionine, and formate (in order of decreasing mean concentration in untreated samples). Some low-concentration compounds were observed to decrease in concentration after treatment, including trimethylamine (-47%), methylamine (-42%) benzoate (-63%), and tyramine (-33%), and lactate was completely eliminated with vancomycin treatment.

7.3.4.2 Defined community

The effects of challenging the resident community with a defined community were also investigated by administering MET-1, a community culture derived from Donor A. The metabolite profile of the Donor A culture remained largely unchanged after the bacterial community was challenged with MET-1 based on PCA of the metabolite profiles of the culture before and after introduction of the defined culture (Figure A9). However, there were still concentration differences apparent in the donor A profile after MET-1 treatment. High-concentration compounds (mean $\geq$ 1 mM) either did not undergo significant changes with MET-1 treatment, or decreased by low-to-moderate amounts. The high-concentration compounds that decreased were propionate (-11%), ethanol (-32%), isobutyrate (-12%), glutamate (-18%) and isovalerate (-7%). Methanol was the only high-concentration compound that exhibited a large decrease in concentration (92%) after MET-1 was introduced. The changes

observed in low-concentration compounds were quite varied; observing both moderate increases and decreases in many compounds. Succinate, which was found at less than 1 mM prior to MET-1 treatment, was that only SCFA that increased with MET-1 treatment, and it did so by more than 100%. Similarly, dimethylamine was the only other low-concentration compound that increased by more than 100% after MET-1 treatment. On the other hand, asparagine, lysine and thymine were all completely eliminated after treatment.

The effects of treatment with a defined culture on a bacterial population in dysbiosis were evaluated by treating the Donor A culture with clindamycin followed by adding MET-1 to the culture. The metabolite profile of the culture after clindamycin and MET-1 treatment was comparable to that of cultures treated only with clindamycin according to the PCA score plot, where compounds were assigned equal weighting (Figure A9). In comparing the treatments' profiles without scaling, it was evident that the concentration changes caused by the combination treatment resembled that of the clindamycin treatment, but with some characteristics of the MET-1 treatment profile (Figure 7-6). Most of the changes observed in high-concentration compounds followed the same trend in the clindamycin-MET-1 combination treatment, as clindamycin-only treatment of Donor A. For some compounds, like acetate and glutamate, the percent change was nearly identical for the combination treatment and clindamycin treatment. For other compounds, like butyrate and valerate, the percent differences was more exaggerated in the combination treatment than the clindamycin treatment, with 42% and 73% decrease, respectively, in the combination treatment, compared to 30% and 24% decrease, respectively, in the clindamycin treatment. There were also compounds that followed the trend of MET-1 treatment, rather than the clindamycin treatment, as exemplified by propionate and isovalerate.

Low-concentration compounds (<1 mM) also contained characteristics from both clindamycin treatment and MET-1 treatment profiles. In fact, about one third of the low-concentration compounds followed the same trends as the changes seen with clindamycin treatment and one-third followed that of MET-1 treatment. The remaining third of the compounds exhibited changes that were unique to the combination treatment, resembling neither the clindamycin treatment nor the MET-1 treatment. Specifically, valine, leucine, and isoleucine did not change significantly when clindamycin was followed with introduction of MET-1 even though those compounds increased with both clindamycin-only and MET-1-only treatments. Glycolate (100+%), serine (40%), threonine (72%) and aspartate (86%) all increased after the combination treatment, but demonstrated no significant changes after clindamycin treatment, and moderate decreases with MET-1 treatment. 3-Phenylpropionate, lactate and malate were also compounds that demonstrated concentration changes after administration of the combination treatment that were different from the changes observed when treated with each constituent perturbation agent alone.

7.3.4.3 Norepinephrine treatment

Exposure of norepinephrine, a stress hormone, to the Donor A-derived culture had a minor impact on the culture's metabolite profiles, as reflected in the PCA of treated and non-treated profiles (Figure A9). In an overview of changes occurring across the entire profile (Figure 7-6), there were very few significant changes, giving further support that little changes occurred due to norepinephrine treatment. The few changes that did occur were observed in low-concentration compounds (mean $\leq$ 1 mM). These changes occurred in succinate (61%), malate (77%), 3-phenylproprionate (100+%), fumarate (100+%) and asparagine (-100%).

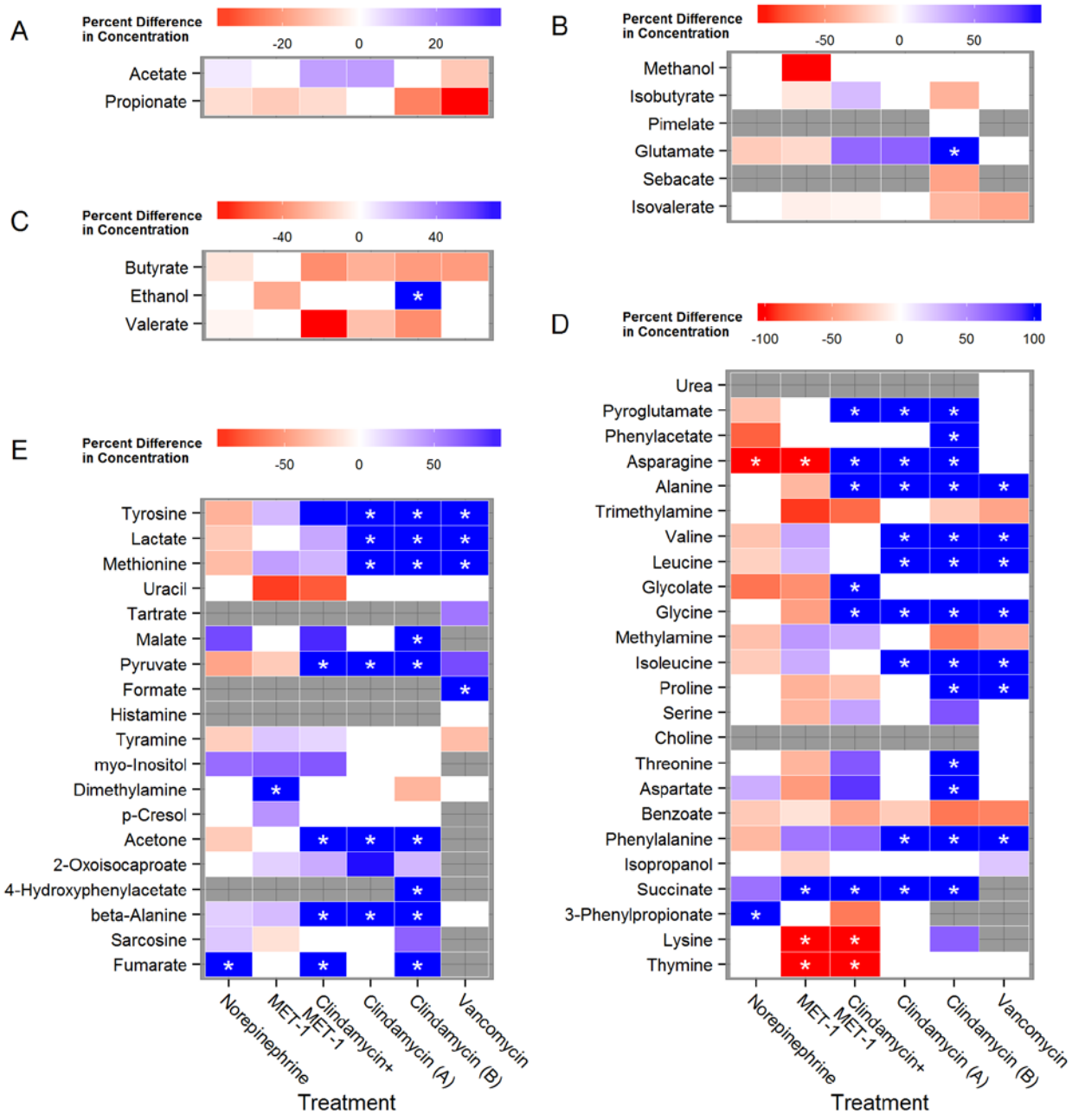

Figure 7-6. Treatment effects on metabolite profiles. The percent difference between mean compound concentration of treated samples and mean compound concentration of their non-perturbed counterparts. Norepinephrine, MET-2, clindamycin-MET-2 combination treatment and clindamycin treatment were administered to bioreactors inoculated with Donor A. Clindamycin treatment was also administered to bioreactors inoculated with Donor B. The two clindamycin conditions are identified by their respective donors as Clindamycin (A) and Clindamycin (B). Vancomycin was administered to a bioreactor growing the defined culture, MET-2. Positive percent differences are represented in blue, negative percent differences are represented in red, and non-significant differences are

represented in white. Compounds not found in either treatment or non-treated conditions for a particular profile are presented grey. Differences exceeding 100% are represented by a star (*) on blue, while differences exceeding -100% are represented by a star on red. Significance of compound changes due to treatment was tested by the Mann-Whitney U Test, where p-value > 0.05 were considered as non-significant. Compounds are organized according to the mean of untreated samples. (A) Bin range: mean $\geq$ 20 mM. (B) Bin range: 1 mM $\leq$ mean < 10 mM. (C) Bin range: 10 mM $\leq$ mean < 20 mM. (D) Bin range: 0.1 mM $\leq$ mean < 1 mM. (E) Bin range: 0 mM $\leq$ mean < 0.1 mM.

7.3.5 Discussion

7.3.5.1 Clindamycin treatment

The impact of clindamycin on the metabolic output of both Donor A and Donor B communities was fairly consistent, with percent changes mirrored in almost every metabolite. The majority of compounds that increased by more than 100% after clindamycin treatment were amino acids. This indicates that pathways involving the metabolism of amino acids were strongly impacted by clindamycin treatment. Indeed, many changes in compound concentrations correspond with known metabolic pathways of anaerobic bacteria. For example, alanine and asparagine all increased more than 100% after clindamycin treatment; glutamate increased by 62.7% in Donor A and more than 100% in Donor B. In *B. fragilis*, the alanine-aspartate-glutamate metabolism pathway is responsible for the breakdown and formation of each of those amino acids[202]. More generally, compounds involved in the glycine-serine-threonine metabolism increased after clindamycin treatment. Glycine, and methionine both increased by more than 100% after antibiotic treatment. Threonine and serine increased significantly in clindamycin treatment of Donor B culture, but no effect was observed in Donor A culture. Valine-leucine-isoleucine biosynthesis was another pathway where all three end products of the pathway increased by more than 100%. Aromatic amino acids such as phenylalanine, and tyrosine also increased. Given that clindamycin acts on the protein synthesis mechanisms of bacteria to inactivate most anaerobic and Gram-positive aerobic bacteria[187,188,203], the increased amounts of free amino acids observed further supports that extensive impairment of amino acid metabolic pathways had occurred following clindamycin administration.

Short chain fatty acids metabolism pathways were also affected by clindamycin treatment. This is apparent in the decreased levels of propionate, butyrate, valerate, isobutyrate and isovalerate. This supports findings from previous reports of decreases in fatty acid output levels of fecal bacteria

in response to clindamycin treatment[204]. Production of short-chain fatty acids occurs due to carbohydrate fermentation by clostridia. Since, in general, members of the genus *Clostridium*, with the exception of *C. difficile*, are susceptible to clindamycin, decreased levels of short-chain fatty acids could be attributed to elimination of organisms that perform carbohydrate fermentation. Interestingly, succinate increased by more than 100% in both donor communities treated with clindamycin. Since succinate is a SCFA closely associated with energy metabolism pathways as well as carbohydrate fermentation in anaerobic bacteria, the large change in succinate levels could be a reflection of drastic shifts in the communities' energy usage[205,206].

7.3.5.2 Vancomycin treatment

Similar to clindamycin treatment, administration of vancomycin caused a reduction in SCFA: propionate, butyrate, and isovalerate. Other SCFAs observed in the metabolite profiles, valerate and isobutyrate, did not change significantly with treatment. This indicated that the antibiotic inactivated the organisms that were metabolizing carbohydrates and producing SCFAs, such as vancomycin-sensitive bacteria from the *Clostridium leptum and Clostridium coccoides* clusters[207]. SCFAs, and lactate are fecal metabolites that have been previously reported to decrease with vancomycin treatment in mice models and human subjects[208,209], even though the findings presented here demonstrated a large increase in lactate. The valine-leucine-isoleucine metabolism pathway as well as phenylalanine-tyrosine-tryptophan pathway were both well represented in the list of amino acids that increased after antibiotic treatment. These amino acid changes are in agreement with previous publications on the metabolomic effects of vancomycin on gut microbiota[208]. Glycine, serine and threonine, which are part of the same metabolism pathway, all increased after vancomycin treatment. Glycine is also involved in energy storage by acting as an oxidizing agent, and it is reduced to acetate and ammonia in one-carbon metabolism pathways[149,210].

7.3.5.3 Challenge of Fecal Culture with Defined Community

The impact of using a defined mixed culture, MET-1, to challenge an already-established fecal culture was investigated. In comparing fecal culture from Donor A before and after it was challenged with MET-1, the metabolite profiles were not impacted strongly by the presence of the defined culture. There were low to moderate decreases in SCFAs that had mean concentrations $\geq$ 1 mM prior to treatment. This was an indication that the Donor A community's ability to ferment carbohydrates was not strongly impacted by the introduction of MET-1. Even though succinate increased by over 100%, such small concentrations were observed that the 100+% increase still only amounted to less than 1 mM. Despite from the small magnitude of change, the doubling of succinate should not be discounted as it could be an indication that MET-1 treatment had implications on Donor A community's energy consumption or conservation pathways[205]. Methanol did not change significantly in all other treatment conditions tested, whereas there was a 91% decrease of methanol with MET-1 treatment. While this was an observation unique to MET-1 treatment, the absolute change in methanol was only 0.7 mM, indicating that the overall impact on the metabolite profile, and the culture as a whole, was very little. Other metabolite-specific changes can be observed with introduction of MET-1, however, like methanol, the effect size of those changes were minimal.

7.3.5.4 Challenge of Fecal Culture in Dysbiosis with Defined Community

MET-1 was also introduced to Donor A fecal culture after it was treated with clindamycin. This was driven by the hypothesis that the defined community would be able to restore a culture in dysbiosis back to the original culture. The metabolite profiles revealed that while MET-1 was not able re-establish the culture back to its original community, the profile of the clindamycin-MET-1 treated culture contained characteristics of the clindamycin-treated Donor A community, the MET-1 treated Donor A community as well as some characteristics unique to the combination treatment. Similar

trends were observed in short-chain fatty acids such as acetate, butyrate, and valerate, indicating that impairment to carbohydrate fermentation pathways by clindamycin persisted even after introduction of MET-1. The increase in amino acids associated with the alanine-glutamate-aspartate pathway and phenylalanine-tyrosine-tryptophan pathway increased, just like the clindamycin-treated cultures. Therefore, certain amino acid degradation pathways impacted by clindamycin treatment were not recovered by the introduction of MET-1.

On the other hand, common features between the MET-1 treatment profile and the combination treatment profile was evidence that some of the metabolic functions of MET-1 were established. For example, decreases in metabolically linked compounds trimethylamine, and methylamine[211,212] observed in both MET-1-treated communities could mean that the pathway involved in the of these compounds became more active with the introduction of MET-1. Similarly, thymine and uracil, both catabolized by *E. coli* as carbon sources in nitrogen-restricted environments, decreased after MET-1 treatment and combination treatment[213]. Some amino acid metabolism in the combination-treated profile was more like the MET-1-treated profile than the clindamycin treated profiles. This included decreases in proline, and lysine, and an increases in methionine. The similar trends that proline and lysine follow in both MET-1 treatment and combination treatment condition suggested that MET-1 established its ability to breakdown nitrogen-rich amino acids. Even though free nitrogen and ammonia levels were not measured, the increase in catabolism of pyrimidines and nitrogen-rich amino acids were evidence that MET-1 conferred its nitrogen-utilisation capabilities to the Donor A community, even when it was perturbed with clindamycin.

As mentioned earlier, the combination-treatment condition displayed some unique characteristics that were not observed in either MET-1 treatment or the combination treatment conditions. In particular, amino acids in the leucine-valine-isoleucine pathway remained the same

after combination treatment, whereas clindamycin-treated only communities exhibited large increases and the MET-1-treated community exhibited moderate increases in those three compounds. The opposing effects of the combination treatment compared to the clindamycin treatment suggested that MET-1 had a remedial effect with respects to the degradation pathway of these amino acids. Furthermore, compounds of the glycine-serine-threonine pathway seemed to be sensitive to difference types of perturbations. Increases in serine and threonine was unique to the combination treatment; methionine increased moderately after both MET-1 treatment and the combination treatment; glycine increased by more than 100% in both combination treatment and clindamycin treatment, but not MET-1 treatment. 3-Phenylprionate decreased by 67% after the combination treatment, an effect not observed with either MET-1 treatment or clindamycin treatment. Since the Donor A community was previously found to harbour phenylalanine degradation capabilities, the decrease in 3-phenylpropionate suggested that the combination treatment impaired the pathway responsible for converting phenylalanine to 3-phenylpropionate, an effect that did not occur with single perturbation treatments. Compounds closely associated with anaerobic energy metabolism, like glycolate, fumarate, lactate, malate and succinate[186,205], also responded to the combination treatment differently than the MET-1 or clindamycin single-treatments. Glycolate, malate and fumarate all increased largely after the combination treatment, but either did not change significantly, or decreased moderately in the single-treatment conditions.

Based on direct comparison of the combination-treated metabolite profiles with other treatment, the signatures of clindamycin treatment profiles were clearly evident. At the same time, it was apparent that MET-1 was able to confer some of its metabolic functions. Interestingly, some of the effects of clindamycin treatment was no longer evident after the combination treatment, suggesting that the MET-1 treatment re-established particular biofunctions that were impaired by the

antibiotic. Since the metabolite profile of the Donor A community after the combination treatment appeared to be a mosaic of clindamycin treatment and MET-1 treatment, it suggested that the community was undergoing a dynamic, transitional stage, where the MET-1 community was in the process of establishing its functional role in the "deteriorated" Donor A community. It was speculated that if the MET-1 community was given more time to assert itself following clindamycin treatment, the metabolite profile of the combination-treated community would become distinct from either clindamycin or MET-1 treatment.

The changes highlighted in the heat maps demonstrated how small changes in the metabolite profiles can yield insight into the changes happening to the community. At the same time, the impact of these changes on the profile as a whole is minimal. As a result, the metabolite profiles of Donor A community after combination treatment and after clindamycin treatment can still be considered to be very similar.

7.3.5.5 Norepinephrine treatment

Previous studies have shown that physiological stress can alter gut microbiota[198]. Norepinephrine, a catecholamine released in response to acute stress, is known to promote the growth of, and expression of virulence determinants by, some Gram-negative enteric bacteria, such as *E. coli* and *Y. enterocolitica*[201], through bacterial signalling pathways. However, based on multivariate analysis the metabolic profiles of a norepinephrine-treated community did not differ notably from the control (untreated) community. A difference in succinate was one of the few changes seen in response to norepinephrine treatment, although this change was not as drastic in comparison to succinate fluctuations seen in response to antibiotic perturbation. One mechanism by which norepinephrine enhances growth of certain enteric organisms involves liberation of glycoprotein-bound iron ions by the catecholamine, rendering iron available to the bacteria to be taken up, promoting their growth. It

is possible that the addition of norepinephrine to the bioreactor system did not have an effect on microbial community growth because iron was supplied in the form of hemin, which is readily taken up by anaerobic bacteria[214,215], and not trapped in a protein-iron complex.

Chapter 8 Conclusions

8.1 Metabolomic analysis of Nutrient Solutions

The targeted profiling technique used to build metabolite profiles of nutrient solutions was able to identify and quantify compounds based on their peak signals in ^{1}H NMR spectra. Despite the manual spectral processing and peak assignment, the technique has been validated to be one that is reproducible and reliable[39,41]. Conducting metabolomic analysis on cell culture media provides insight into the nutrient content of the growth solution. Therefore, as a demonstration of the metabolomic analysis in the context of nutrient solutions, the metabolite make up of a defined media for mammalian CHO cells subjected to various treatments of UV irradiation was investigated. The findings from this study revealed that not only was the constructed metabolite profiles a comprehensive annotation of the nutrient solution, the profiles were also sensitive enough to quantify the minimal effects of UV treatment. Furthermore, this experiment demonstrated that the metabolite profiling technique would be an appropriate tool in observing changes in low-molecular weight compounds found in cell culture media. Therefore, the application of this technique can be expanded to investigating metabolites of spent media, from the growth of bacterial communities, such as those of the human gut microbiota.

8.2 Standardization of Reporting on Fecal Water Metabolites

Metabolomic studies of the human gut microbiota thus far have focused heavily on fecal samples. However, the minimal degree of standardization of obtaining NMR-based metabolomic data of feces[62] and the complete lack of standardization in reporting the fecal metabolomic data have prevented cross-experimental comparisons. Therefore, with the purpose of determining an appropriate means of reporting fecal metabolome, various concentrations of human fecal water were analyzed. NMR-based analysis of these fecal water samples revealed that there was a direct linear relationship

between fecal concentration and metabolite concentration. The implication of this is that the concentration of biomass used in preparing such samples must be well documented and transparent to all experiment stakeholders, or be reported as normalized values. Two methods of normalizing metabolite profiles were investigated: one where the concentrations were scaled to the mass of feces from which the metabolites were extracted, and the other, where the concentrations were scaled to an internal compound that was observed in abundance. While both normalization methods were equally effective, they both had advantages and disadvantages in their application. Furthermore, given the linear relationship between metabolite concentration and fecal concentration, the most suitable fecal-water concentration would be defined by the intended analysis method. For the purposes of NMR-based metabolite concentration, fecal concentration should be greater than 25% fecal concentration in order to ensure low concentration compounds are captured within the limit of quantification. Alternatively, it can be expressed that fecal water samples must contain at least ~7.3 mM of acetate to construct accurate and encompassing metabolite profiles. Beyond analytical considerations, practical limitations of sample preparation should also be taken into account. Since all samples must be passed through a 0.22 um filter, this may serve as an upper limit to the concentration of fecal water that can be processed.

Based on these findings, it is recommended that acetate be used as a proxy to qualifying the contents of a fecal water sample. Acetate is an attractive identifying marker because it has been consistently found to be the most abundant compound in fecal water samples. According to the samples evaluated here, fecal water samples should be evaluated to contain at least 7.3 mM or 0.43 g/L of acetate in order to qualify for metabolomic analysis by 1D-^{1}H NMR. Describing a sample in terms of acetate content could be a more standardized method of describing fecal water samples. While 7.3 mM of acetate was the minimum requirement determined from this experiment, further

study on other stool samples would be required to determine a generalized acetate threshold for fecal water samples.

8.3 Metabolomic Evaluation of *In Vitro* Gut Bacterial Communities

A wealth of insight has been gained on the human gut microbiota through the use of *in vitro* growth of the gut microbes as a community culture. These *in vitro* systems have been validated for their ability to preserve the *in vivo* gut bacterial community on the microbiological level[11,68,75], but little work has been put forth to determine the metabolomic implications of these *in vitro* gut microbial cultures. In an effort to provide such metabolomic descriptions of *in vitro* gut bacterial communities, two experiments were conducted. The first centered on developing an appropriate procedure for preparing bioreactor samples for NMR-based metabolomic analysis. The second investigated the metabolomic impact of the microbial make-up of the fecal inoculant on its resulting *in vitro* bacterial community.

8.3.1 Sample Preparation Procedure for Metabolomic Analysis

Comparison of various procedures for the preparation of bioreactor samples for NMR-based metabolomic analysis determined that factors such as performing a “soft spin” to remove cell matter and syringe filtration had no significant effect on the metabolite profiles. On the other hand, the centrifugal force impacted the metabolite profile. It was found that increasing centrifugal force applied the effluent samples generally decreased compound concentrations in a linear fashion. These linear decreases, while significant, only occurred within 2% of the mean compound concentration and can be considered to be of little importance. The only compound to increase in concentration with centrifugal force applied was p-cresol, which increased by approximately 50% per 10 000 rpm applied. Furthermore, these effects were found to occur with high reproducibility. In conclusion,

centrifugal force had significant, but minimal impact on the metabolite profile of the sample, but additional attention should be allocated to p-cresol. The large and predictable means by which this compound is impacted by ultracentrifugation means that a correction factor of 50% per 10 000 rpm should be applied to the compound's concentration. Compensation for other compounds is not recommended as their trends were not observed at as high precision, and changes made to their concentrations due to sample processing steps are so small as to be considered negligible.

8.3.2 Effects of Varying Inoculant Biomass Proportions

There were four different community cultures evaluated in this experiment: two different defined cultures, MET-2A and MET-3A and their respective reciprocal cultures, MET-2B and MET-3B. The reciprocal cultures consisted of the same strains as the parent culture (MET-2A or MET-3A), but with different proportions of each strain in the inoculum. The objective of this experiment was to observe the metabolite profiles of the culture over time in a CSTR system and determine if similar metabolic states would be achieved regardless of inioculant biomass. After determining sources of variation to be mostly attributed to different inocula, rather than experimental parameters such as run variability or profiling variability, the effect of varying inoculum biomass was investigated. First of all, the total concentration of metabolites in a defined culture was similar to the total metabolite concentration in the profiles of the respective reciprocal population. Evaluation of the cultures over time revealed that the majority of compounds appeared to approach a common level by day 20 when comparing the MET-2A and MET-3A with their respective reciprocal cultures. In order to draw more accurate conclusions of the trends in metabolite profile changes over time, however, metabolite profiles should be assembled for more time points throughout the duration of the culture, with multiple samples per time point for an estimate of error and variation and also for a longer duration in order to evalutate acheivement of metabolic steady state.

8.4 Metabolomic Characterization of In *Vitro* Gut Microbial Communities

The advantages associated with studying the gut microbiota in an *in vitro* system combined with the metabolomic data available in these systems allows for a wide range of studies that contribute to metabolomic characterization the human gut microbiota. Therefore, two experiments were conducted with the aim of providing further insight into the metabolome of these bacterial communities. In "Metabolomic Analysis of Human Fecal Microbiota Propagated *In Vitro* in a CSTR System," it was found that the metabolite profiles of the gut bacterial communities were generally composed of the same proportion of classifications of metabolites, regardless of the individual from whom the inoculating community was obtained. Despite these general similarities, the metabolite profiles were still able to distinguish one bacterial community from the next, even identifying differences in metabolic capabilities. These metabolomic characterizations were also applied to a defined community cultured in the same *in vitro* bioreactor system, revealing some functional differences between feces-derived and defined culture communities.

Given that the metabolite profiles were able to identify individual bacterial communities and comment on their metabolic functions, it was of interest to observe the *in vitro* communities' responses to various perturbations. For this purpose, several perturbation agents were administered to feces-derived or defined communities and metabolite profiles were used to identify the changes undergone by the cultures. These treatments included two antibiotic treatments (clindamycin and vancomycin); challenging the resident community with a defined community, MET-2; an antibiotic-MET-2 combination treatment; and a norepinephrine treatment. It was found that amino acid and carbohydrate metabolism were the most sensitive to perturbation, as various changes in levels of associated amino acids and fatty acids were observed in all treatments except for norepinephrine treatment, which did not drastically impact the community's metabolite profile. Succinate was a

notable compound as it increased considerably after every treatment, even norepinephrine treatment. Due to succinate's close association with central metabolic processes and is subject to large changes with perturbation, it could be the first indicator of a perturbation to the culture. From this study, it has been demonstrated that metabolite profiles of mixed cultures allows the differentiation of communities of different inoculant sources. These metabolite profiles also allow characterization of various perturbations, such as antibiotics, probiotic treatment and stress exposure, on those communities.

The complex symbiotic relationship between microbial ecosystems residing in the human GI tract and human health emphasizes the importance of developing a comprehensive understanding of the interactions through alternative and complementary approaches. Studying the metabolome of the gut microbes is one such approach that has the capability of deciphering those interactions. In this body of work, an *in vitro* system of studying the human gut microbiota was subjected to metabolomic evaluation. Standardizing reports on fecal metabolomes and determining source of variation in metabolomic analyses are fundamental characterizations that appropriately contextualizes the findings presented here so they can be effectively communicated to other researchers. The metabolomic approach was further applied in analyzing the exometabolome found in the nutrient solutions in which these community cultures were grown. The metabolite profiles defined microbial metabolite products that commented on the summative outcome of the complex microbial interactions occurring within the gut microbiota. This is relevant to clinical and industrial research because this metabolomic characterization has the potential to bridge microbial activities with human health and physiology.

Chapter 9 Recommendations

The insights ascertained in this thesis contribute to the metabolomic understanding of the human gut microbiota by setting foundational work from which this approach can be further developed. For example, further metabolomic evaluation of the single-stage CSTR system can be performed. Comparison of the *in vitro* communities derived from two inocula that differed only in biomass proportions of the same bacterial species demonstrated that communities grown in the bioreactor systems were relatively robust, with the final ecological consortium driven by metabolic function, rather than the proportional composition of the inoculant. This finding suggested that the metabolic steady state was being approached as of 20 days post-inoculation. Some fluctuation in the metabolite levels indicated that the system was still dynamic, but convergence of profiles between a culture (MET-2A or MET-3A) and its reciprocal variant (MET-2B or MET-3B) was a hint that metabolic interactions were becoming more stable. Therefore, it would be worthwhile to conduct an experiment in which a metabolomic time-trend can establish metabolic steady state is achieved and how this compares to when ecological steady state is reached.

There is an evident gap in human gut microbial research, where advanced *in vitro* systems have been developed to study the ecology of the human gut microbiota, and at the same time, metabolomic understanding of the human gut microbiota have focused on fecal metabolites. A direct comparison of the fecal metabolome with the metabolome of the *in vitro* community cultured from the same fecal sample would be a good depiction of how the fecal communities change or are preserved in the *in vitro* system. This analysis would be a starting point for bridging that gap between *in vitro* gut microbiota research and our metabolomic understanding of that ecosystem.

The metabolite profile analysis of feces-derived communities and defined communities was able to confer a wealth of information regarding the gut microbiota. Not only did it highlight gross

differences between each community, such as decreased levels carbohydrate fermentation in the defined community compared to its fecal counterpart, but specialized metabolic functions were identified to be unique to certain communities. Since the metabolomic data was able to divulge details on the metabolic interactions in the bacterial consortia, this approach can be applied to studying the human gut microbiota in clinical or pharmaceutical contexts. A wide range of perturbations were administered to the gut as a demonstration of such applications. From this preliminary experiment, it was demonstrated that amino acid metabolism and carbohydrate fermentation were the most dramatically influenced by antibiotic treatments. It is worth noting that the antibiotics used were administered to different kinds of bacterial communities (feces-derived or defined). Given that clindamycin and vancomycin targeted cells through different mechanisms of action – clindamycin inhibits the ribosomal mechanisms to halt protein interaction, while vancomycin interrupts biosynthesis of bacterial cell walls – it would be interesting to see how different classes of antibiotics impact one bacterial consortium.

Furthermore, challenging a feces-derived community with a defined community and the combination treatment of clindamycin followed by challenging with a defined community also yielded interesting results. While the clindamycin-only treatment and the defined community-only treatment both produced unique metabolomic responses in the bacterial community, the combination treatment elicited a metabolite profile that resembled a hybrid of the two stand-alone treatments. This experiment would benefit from extending the time during which the microbial responses are monitored. In observing the *in vitro* cultures until steady-state is reached again, the full, long-term impact of the treatments, whether it be stand-alone or combination treatments, on the metabolic and ecologic makeup of the community would be captured. Additionally, the effect of recovery time on the final outcome in the combination treatment should be investigated. Such a study would reveal

whether the stability of the community impacts the ability of the challenging consortia to establish itself within the residing community.

The norepinephrine treatment was an example of using the *in vitro* communities to investigate host physiology on the gut microbiota. Even though host stress response have been show to impact bacterial physiology and the health of gut microbiota as a whole, those interactions were not reproduced with the when the *in vitro* feces-derived community was treated with norepinephrine. The reason for this discrepancy was hypothesized to be attributed the abundance of iron supplied in the media feed. Since iron was readily available throughout the culturing process, norepinephrine did not have any added benefit to the microbes that would typically capitalize on the norepinephrine-facilitated release on iron ions was lost.

The work presented in this thesis provides a kernel of insight into the contributions metabolomic studies could have on understanding the human gut microbiota. The recommendations stemming from this work focus on validating the bioreactor systems involved in propagating gut microbes *in vitro* as well as in deciphering metabolic interactions of those communities.

Refrences

(1) Mapelli, V., Olsson, L., and Nielsen, J. (2008) Metabolic footprinting in microbiology: methods and applications in functional genomics and biotechnology. *Trends Biotechnol. 26*, 490–7.

(2) Saric, J., Wang, Y., Li, J., and Coen, M. (2007) Species variation in the fecal metabolome gives insight into differential gastrointestinal function. *J. Proteome Res. 7*, 352–360.

(3) Le Gall, G., Noor, S. O., Ridgway, K., Scovell, L., Jamieson, C., Johnson, I. T., Colquhoun, I. J., Kemsley, E. K., and Narbad, A. (2011) Metabolomics of fecal extracts detects altered metabolic activity of gut microbiota in ulcerative colitis and irritable bowel syndrome. *J. Proteome Res. 10*, 4208–18.

(4) Jacobsen, N. (2007) NMR spectroscopy explained: simplified theory, applications and examples for organic chemistry and structural biology, pp 353–450.

(5) McDonald, J. a K., Schroeter, K., Fuentes, S., Heikamp-Dejong, I., Khursigara, C. M., de Vos, W. M., and Allen-Vercoe, E. (2013) Evaluation of microbial community reproducibility, stability and composition in a human distal gut chemostat model. *J. Microbiol. Methods 95*, 167–74.

(6) Kim, B.-S., Kim, J. N., and Cerniglia, C. E. (2011) In vitro culture conditions for maintaining a complex population of human gastrointestinal tract microbiota. *J. Biomed. Biotechnol. 2011*, 838040.

(7) Fiehn, O., Kopka, J., Dörmann, P., Altmann, T., Trethewey, R. N., and Willmitzer, L. (2000) Metabolite profiling for plant functional genomics 1157–1161.

(8) Nicholson, J. K., Lindon, J. C., and Holmes, E. (1999) "Metabonomics": understanding the metabolic responses of living systems to pathophysiological stimuli via multivariate statistical analysis of biological NMR spectroscopic data. *Xenobiotica. 29*, 1181–9.

(9) Phelps, T. J., Palumbo, A. V, and Beliaev, A. S. (2002) Metabolomics and microarrays for improved understanding of phenotypic characteristics controlled by both genomics and environmental constraints. *Curr. Opin. Biotechnol. 13*, 20–24.

(10) Fiehn, O. (2002) Metabolomics--the link between genotypes and phenotypes. *Plant Mol. Biol. 48*, 155–71.

(11) Madsen, R., Lundstedt, T., and Trygg, J. (2010) Chemometrics in metabolomics--a review in human disease diagnosis. *Anal. Chim. Acta 659*, 23–33.

(12) Gibney, M. J., Walsh, M., Brennan, L., Roche, H. M., German, B., and van Ommen, B. (2005) Metabolomics in human nutrition: opportunities and challenges. *Am. J. Clin. Nutr. 82*, 497–503.

(13) Mashego, M. R., Rumbold, K., De Mey, M., Vandamme, E., Soetaert, W., and Heijnen, J. J. (2007) Microbial metabolomics: past, present and future methodologies. *Biotechnol. Lett. 29*, 1–16.

(14) Vigneau-Callahan, K. E., Shestopalov, A. I., Milbury, P. E., Matson, W. R., and Kristal, B. S. (2001) Characterization of diet-dependent metabolic serotypes: analytical and biological variability issues in rats. *J. … 131*, 924S–932S.

(15) Gross, J. H. (2014) Mass Spectrometry 2nd ed., pp 355–406. Springer, Heidlberg.

(16) Takahashi, H., Kai, K., Shinbo, Y., Tanaka, K., Ohta, D., Oshima, T., Altaf-Ul-Amin, M., Kurokawa, K., Ogasawara, N., and Kanaya, S. (2008) Metabolomics approach for determining growth-specific metabolites based on Fourier transform ion cyclotron resonance mass spectrometry. *Anal. Bioanal. Chem. 391*, 2769–82.

(17) Bristow, A. W. T., Webb, K. S., Lubben, A. T., and Halket, J. (2004) Reproducible product-ion tandem mass spectra on various liquid chromatography/mass spectrometry instruments for the development of spectral libraries. *Rapid Commun. Mass Spectrom. 18*, 1447–54.

(18) De Vos, R. C. H., Moco, S., Lommen, A., Keurentjes, J. J. B., Bino, R. J., and Hall, R. D. (2007) Untargeted large-scale plant metabolomics using liquid chromatography coupled to mass spectrometry. *Nat. Protoc. 2*, 778–91.

(19) Lisec, J., Schauer, N., Kopka, J., Willmitzer, L., and Fernie, A. R. (2006) Gas chromatography mass spectrometry-based metabolite profiling in plants. *Nat. Protoc. 1*, 387–96.

(20) Gao, X., Pujos-Guillot, E., Martin, J.-F., Galan, P., Juste, C., Jia, W., and Sebedio, J.-L. (2009) Metabolite analysis of human fecal water by gas chromatography/mass spectrometry with ethyl chloroformate derivatization. *Anal. Biochem. 393*, 163–75.

(21) Gao, X., Pujos-Guillot, E., and Sébédio, J.-L. (2010) Development of a quantitative metabolomic approach to study clinical human fecal water metabolome based on trimethylsilylation derivatization and GC/MS analysis. *Anal. Chem. 82*, 6447–56.

(22) Dumas, M.-E., Maibaum, E. C., Teague, C., Ueshima, H., Zhou, B., Lindon, J. C., Nicholson, J. K., Stamler, J., Elliott, P., Chan, Q., and Holmes, E. (2006) Assessment of

analytical reproducibility of 1H NMR spectroscopy based metabonomics for large-scale epidemiological research: the INTERMAP Study. *Anal. Chem. 78*, 2199–208.

(23) Beckonert, O., Keun, H. C., Ebbels, T. M. D., Bundy, J., Holmes, E., Lindon, J. C., and Nicholson, J. K. (2007) Metabolic profiling, metabolomic and metabonomic procedures for NMR spectroscopy of urine, plasma, serum and tissue extracts. *Nat. Protoc. 2*, 2692–703.

(24) Wevers, R. A., Engelke, U., and Heerschap, A. (1994) High-Resolution 1HNMR Spectroscopy of Blood Plasma for Metabolic Studies *40*, 1245–1250.

(25) Monakhova, Y. B., Schäfer, H., Humpfer, E., Spraul, M., Kuballa, T., and Lachenmeier, D. W. (2011) Application of automated eightfold suppression of water and ethanol signals in 1H NMR to provide sensitivity for analyzing alcoholic beverages. *Magn. Reson. Chem. 49*, 734–9.

(26) Hwang, T., and Shaka, A. (1995) Water suppression that works. Excitation sculpting using arbitrary wave-forms and pulsed-field gradients. *J. Magn. Reson. Ser. A 112*, 275–279.

(27) Viant, M. R. (2003) Improved methods for the acquisition and interpretation of NMR metabolomic data. *Biochem. Biophys. Res. Commun. 310*, 943–948.

(28) Tang, H., Wang, Y., Nicholson, J. K., and Lindon, J. C. (2004) Use of relaxation-edited one-dimensional and two dimensional nuclear magnetic resonance spectroscopy to improve detection of small metabolites in blood plasma. *Anal. Biochem. 325*, 260–272.

(29) Chang, D., Banack, C. D., and Shah, S. L. (2007) Robust baseline correction algorithm for signal dense NMR spectra. *J. Magn. Reson. 187*, 288–92.

(30) Jacob, D., Deborde, C., and Moing, A. (2013) An efficient spectra processing method for metabolite identification from (1)H-NMR metabolomics data. *Anal. Bioanal. Chem.*

(31) Weljie, A. M., Newton, J., Mercier, P., Carlson, E., and Slupsky, C. M. (2006) Targeted profiling: quantitative analysis of 1H NMR metabolomics data. *Anal. Chem. 78*, 4430–42.

(32) Reynolds, W., and Enriquez, R. (2002) Choosing the best pulse sequences, acquisition parameters, postacquisition processing strategies, and probes for natural product structure elucidation by NMR. *J. Nat. Prod. 65*, 221–244.

(33) Beckwith-Hall, B. M., Nicholson, J. K., Nicholls, A. W., Foxall, P. J. D., Lindon, J. C., Connor, S. C., Abdi, M., Connelly, J., and Holmes, E. (1998) Nuclear Magnetic Resonance Spectroscopic and Principal Components Analysis Investigations into Biochemical Effects of Three Model Hepatotoxins. *Chem. Res. Toxicol. 11*, 260–272.

(34) Holmes, E., and Antti, H. (2002) Chemometric contributions to the evolution of metabonomics: mathematical solutions to characterising and interpreting complex biological NMR spectra. *Analyst 127*, 1549–1557.

(35) Penha, R., and Hines, J. (2001) Using principal component analysis modeling to monitor temperature sensors in a nuclear research reactor. *Proc. Maint. Reliab.* … 6–9.

(36) Jolliffe, I. T. (2002) Principal Component Analysis, Second Edition 2nd ed., pp 1–61. Springer-Verlag New York, Inc.

(37) Duarte, I. F., Lamego, I., Rocha, C., and Gil, A. M. (2009) NMR metabonomics for mammalian cell metabolism studies. *Bioanalysis 1*, 1597–614.

(38) Lindon, J., and Nicholson, J. (2000) Metabonomics: metabolic processes studied by NMR spectroscopy of biofluids. *Concepts Magn. Reson. 12*, 289–320.

(39) Sokolenko, S., Blondeel, E., Azlah, N., George, B., Schulze, S., Chang, D., and Aucoin, M. G. (2014) Profiling convoluted single-dimension proton NMR spectra: A Plackett–Burman approach for assessing quantification error of metabolites in complex mixtures with application to cell culture. *Anal. Chem. 86*, 3330–3337.

(40) Tredwell, G. D., Behrends, V., Geier, F. M., Liebeke, M., and Bundy, J. G. (2011) Between-person comparison of metabolite fitting for NMR-based quantitative metabolomics. *Anal. Chem. 83*, 8683–7.

(41) Sokolenko, S., McKay, R., Blondeel, E. J. M., Lewis, M. J., Chang, D., George, B., and Aucoin, M. G. (2013) Understanding the variability of compound quantification from targeted profiling metabolomics of 1D-1H-NMR spectra in synthetic mixtures and urine with additional insights on choice of pulse sequences and robotic sampling. *Metabolomics 9*, 887–903.

(42) Kaderbhai, N. N., Broadhurst, D. I., Ellis, D. I., Goodacre, R., and Kell, D. B. (2003) Functional genomics via metabolic footprinting: monitoring metabolite secretion by Escherichia coli tryptophan metabolism mutants using FT-IR and direct injection electrospray mass spectrometry. *Comp. Funct. Genomics 4*, 376–91.

(43) Piper, P. (1995) The heat shock and ethanol stress responses of yeast exhibit extensive similarity and functional overlap. *FEMS Microbiol. Lett. 134*, 121–127.

(44) Devantier, R., Scheithauer, B., Villas-Bôas, S. G., Pedersen, S., and Olsson, L. (2005) Metabolite profiling for analysis of yeast stress response during very high gravity ethanol fermentations. *Biotechnol. Bioeng. 90*, 703–14.

(45) Aranibar, N., Borys, M., Mackin, N. a, Ly, V., Abu-Absi, N., Abu-Absi, S., Niemitz, M., Schilling, B., Li, Z. J., Brock, B., Russell, R. J., Tymiak, A., and Reily, M. D. (2011) NMR-based metabolomics of mammalian cell and tissue cultures. *J. Biomol. NMR 49*, 195–206.

(46) Bradley, S. a, Ouyang, A., Purdie, J., Smitka, T. a, Wang, T., and Kaerner, A. (2010) Fermentanomics: monitoring mammalian cell cultures with NMR spectroscopy. *J. Am. Chem. Soc. 132*, 9531–3.

(47) Read, E. K., Bradley, S. a, Smitka, T. a, Agarabi, C. D., Lute, S. C., and Brorson, K. a. (2013) Fermentanomics informed amino acid supplementation of an antibody producing mammalian cell culture. *Biotechnol. Prog. 29*, 745–53.

(48) Khetan, A., Huang, Y., Dolnikova, J., Pederson, N. E., Wen, D., Yusuf-Makagiansar, H., Chen, P., and Ryll, T. (2010) Control of misincorporation of serine for asparagine during antibody production using CHO cells. *Biotechnol. Bioeng. 107*, 116–23.

(49) Qin, J., Li, R., Raes, J., Arumugam, M., Burgdorf, K. S., Manichanh, C., Nielsen, T., Pons, N., Levenez, F., Yamada, T., Mende, D. R., Li, J., Xu, J., Li, S., Li, D., Cao, J., Wang, B., Liang, H., Zheng, H., Xie, Y., Tap, J., Lepage, P., Bertalan, M., Batto, J.-M., Hansen, T., Le Paslier, D., Linneberg, A., Nielsen, H. B., Pelletier, E., Renault, P., Sicheritz-Ponten, T., Turner, K., Zhu, H., Yu, C., Li, S., Jian, M., Zhou, Y., Li, Y., Zhang, X., Li, S., Qin, N., Yang, H., Wang, J., Brunak, S., Doré, J., Guarner, F., Kristiansen, K., Pedersen, O., Parkhill, J., Weissenbach, J., Bork, P., Ehrlich, S. D., and Wang, J. (2010) A human gut microbial gene catalogue established by metagenomic sequencing. *Nature 464*, 59–65.

(50) Human, T., and Project, M. (2012) Structure, function and diversity of the healthy human microbiome. *Nature 486*, 207–14.

(51) Arumugam, M., Raes, J., Pelletier, E., Le Paslier, D., Yamada, T., Mende, D. R., Fernandes, G. R., Tap, J., Bruls, T., Batto, J.-M., Bertalan, M., Borruel, N., Casellas, F., Fernandez, L., Gautier, L., Hansen, T., Hattori, M., Hayashi, T., Kleerebezem, M., Kurokawa, K., Leclerc, M., Levenez, F., Manichanh, C., Nielsen, H. B., Nielsen, T., Pons, N., Poulain, J., Qin, J., Sicheritz-Ponten, T., Tims, S., Torrents, D., Ugarte, E., Zoetendal, E. G., Wang, J., Guarner, F., Pedersen, O., de Vos, W. M., Brunak, S., Doré, J., Antolín, M., Artiguenave, F., Blottiere, H. M., Almeida, M., Brechot, C., Cara, C., Chervaux, C., Cultrone, A., Delorme, C., Denariaz, G., Dervyn, R., Foerstner, K. U., Friss, C., van de Guchte, M., Guedon, E., Haimet, F., Huber, W., van Hylckama-Vlieg, J., Jamet, A., Juste, C., Kaci, G., Knol, J., Lakhdari, O., Layec, S., Le Roux, K., Maguin, E., Mérieux, A., Melo Minardi, R., M'rini, C., Muller, J., Oozeer, R., Parkhill, J., Renault, P., Rescigno, M., Sanchez, N., Sunagawa, S., Torrejon, A., Turner, K., Vandemeulebrouck, G., Varela, E., Winogradsky, Y., Zeller, G., Weissenbach, J., Ehrlich, S. D., and Bork, P. (2011) Enterotypes of the human gut microbiome. *Nature 473*, 174–80.

(52) Yatsunenko, T., Rey, F. E., Manary, M. J., Trehan, I., Dominguez-Bello, M. G., Contreras, M., Magris, M., Hidalgo, G., Baldassano, R. N., Anokhin, A. P., Heath, A. C., Warner, B., Reeder, J., Kuczynski, J., Caporaso, J. G., Lozupone, C. a, Lauber, C., Clemente, J. C., Knights, D., Knight, R., and Gordon, J. I. (2012) Human gut microbiome viewed across age and geography. *Nature 486*, 222–7.

(53) Li, M., Wang, B., Zhang, M., Rantalainen, M., Wang, S., Zhou, H., Zhang, Y., Shen, J., Pang, X., Zhang, M., Wei, H., Chen, Y., Lu, H., Zuo, J., Su, M., Qiu, Y., Jia, W., Xiao, C., Smith, L. M., Yang, S., Holmes, E., Tang, H., Zhao, G., Nicholson, J. K., Li, L., and Zhao, L. (2008) Symbiotic gut microbes modulate human metabolic phenotypes. *Proc. Natl. Acad. Sci. U. S. A. 105*, 2117–22.

(54) Martin, F., and Sprenger, N. (2010) Dietary modulation of gut functional ecology studied by fecal metabonomics. *J. Proteome Res. 9*, 5284–5295.

(55) Jacobs, D. M., Deltimple, N., van Velzen, E., van Dorsten, F. a, Bingham, M., Vaughan, E. E., and van Duynhoven, J. (2008) (1)H NMR metabolite profiling of feces as a tool to assess the impact of nutrition on the human microbiome. *NMR Biomed. 21*, 615–26.

(56) Shinohara, K., Ohashi, Y., Kawasumi, K., Terada, A., and Fujisawa, T. (2010) Effect of apple intake on fecal microbiota and metabolites in humans. *Anaerobe 16*, 510–5.

(57) Cross, A. J., Greetham, H. L., Pollock, J. R. A., Rowland, I. R., and Bingham, S. A. (2006) Variability in fecal water genotoxicity, determined using the comet assay, is independent of endogenous N-nitroso compound formation attributed to red meat consumption. *Environ. Mol. Mutagen. 47*, 179–184.

(58) Jansson, J., Willing, B., Lucio, M., Fekete, A., Dicksved, J., Halfvarson, J., Tysk, C., and Schmitt-Kopplin, P. (2009) Metabolomics reveals metabolic biomarkers of Crohn's disease. *PLoS One 4*, e6386.

(59) Monleón, D., Morales, J. M., Barrasa, A., López, J. A., Vázquez, C., and Celda, B. (2009) Metabolite profiling of fecal water extracts from human colorectal cancer. *NMR Biomed. 22*, 342–8.

(60) Ponnusamy, K., Choi, J. N., Kim, J., Lee, S.-Y., and Lee, C. H. (2011) Microbial community and metabolomic comparison of irritable bowel syndrome faeces. *J. Med. Microbiol. 60*, 817–27.

(61) Ikeda, N., Saito, Y., Shimizu, J., Ochi, a, Mizutani, J., and Watabe, J. (1994) Variations in concentrations of bacterial metabolites, enzyme activities, moisture, pH and bacterial composition between and within individuals in faeces of seven healthy adults. *J. Appl. Bacteriol. 77*, 185–94.

(62) Wu, J., An, Y., Yao, J., Wang, Y., and Tang, H. (2010) An optimised sample preparation method for NMR-based faecal metabonomic analysis. *Analyst 135*, 1023–30.

(63) Zheng, X., Xie, G., Zhao, A., Zhao, L., Yao, C., Chiu, N. H. L., Zhou, Z., Bao, Y., Jia, W., Nicholson, J. K., and Jia, W. (2011) The footprints of gut microbial-mammalian co-metabolism. *J. Proteome Res. 10*, 5512–22.

(64) Antunes, L. C. M., Han, J., Ferreira, R. B. R., Lolić, P., Borchers, C. H., and Finlay, B. B. (2011) Effect of antibiotic treatment on the intestinal metabolome. *Antimicrob. Agents Chemother. 55*, 1494–503.

(65) Martin, F.-P. J., Wang, Y., Sprenger, N., Yap, I. K. S., Lundstedt, T., Lek, P., Rezzi, S., Ramadan, Z., van Bladeren, P., Fay, L. B., Kochhar, S., Lindon, J. C., Holmes, E., and Nicholson, J. K. (2008) Probiotic modulation of symbiotic gut microbial-host metabolic interactions in a humanized microbiome mouse model. *Mol. Syst. Biol. 4*, 157.

(66) Wang, X., and Gibson, G. (1993) Effects of the in vitro fermentation of oligofructose and inulin by bacteria growing in the human large intestine. *J. Appl. Bacteriol. 75*, 373–380.

(67) Aura, A.-M., Oikarinen, S., Mutanen, M., Heinonen, S.-M., Adlercreutz, H. C. T., Virtanen, H., and Poutanen, K. S. (2006) Suitability of a batch in vitro fermentation model using human faecal microbiota for prediction of conversion of flaxseed lignans to enterolactone with reference to an in vivo rat model. *Eur. J. Nutr. 45*, 45–51.

(68) Campbell, W. L., Franklin, W., and Cerniglia, C. E. (1992) Validation studies on an in vitro semicontinuous culture system designed to simulate a bacterial ecosystem of the human intestine. *J. Microbiol. Methods 16*, 239–252.

(69) Allison, C., Mcfarlan, C., and Macfarlane, G. T. (1989) Studies on mixed populations of human intestinal bacteria grown in single-stage and multistage continuous culture systems. *Appl. Environ. Microbiol. 55*, 672–678.

(70) Gibson, G. R., and Roberfroid, M. B. (1995) Dietary modulation of the human colonie microbiota: introducing the concept of prebiotics. *J. Nutr. 125*, 1401–1412.

(71) Fuller, R. (1989) Probiotics in man and animals. *J. Appl. Bacteriol. 66*, 365–378.

(72) Duncan, S., Scott, K., Ramsay, A. G., Harmsen, H. J., Welling, G. W., Stewart, C. S., and Flint, H. J. (2003) Effects of alternative dietary substrates on competition between human colonic bacteria in an anaerobic fermentor system. *Appl. Environ. Microbiol. 69*, 1139–1142.

(73) Steer, T. E., Johnson, I. T., Gee, J. M., and Gibson, G. R. (2007) Metabolism of the soyabean isoflavone glycoside genistin in vitro by human gut bacteria and the effect of prebiotics. *Br. J. Nutr. 90*, 635.

(74) Bearne, C. a, Mallett, a K., Rowland, I. R., and Brennan-Craddock, W. E. (1990) Continuous culture of human faecal bacteria as an in vitro model for the colonic microflora. *Toxicol. In Vitro 4*, 522–5.

(75) Macfarlane, G., Macfarlane, S., and Gibson, G. (1998) Validation of a Three-Stage Compound Continuous Culture System for Investigating the Effect of Retention Time on the Ecology and Metabolism of Bacteria in the Human Colon. *Microb. Ecol. 35*, 180–7.

(76) Molly, K., Woestyne, M., and Verstraete, W. (1993) Development of a 5-step multi-chamber reactor as a simulation of the human intestinal microbial ecosystem. *Appl. Microbiol. Biotechnol. 1*, 254–258.

(77) Nakamura, N., Lin, H. C., McSweeney, C. S., Mackie, R. I., and Gaskins, H. R. (2010) Mechanisms of microbial hydrogen disposal in the human colon and implications for health and disease. *Annu. Rev. Food Sci. Technol. 1*, 363–95.

(78) Feria-Gervasio, D., Denis, S., Alric, M., and Brugère, J.-F. (2011) In vitro maintenance of a human proximal colon microbiota using the continuous fermentation system P-ECSIM. *Appl. Microbiol. Biotechnol. 91*, 1425–33.

(79) Feria-Gervasio, D., Tottey, W., Gaci, N., Alric, M., Cardot, J.-M., Peyret, P., Martin, J.-F., Pujos, E., Sébédio, J.-L., and Brugère, J.-F. (2014) Three-stage continuous culture system with a self-generated anaerobia to study the regionalized metabolism of the human gut microbiota. *J. Microbiol. Methods 96*, 111–8.

(80) Minekus, M., Smeets-Peeters, M., Bernalier, a., Marol-Bonnin, S., Havenaar, R., Marteau, P., Alric, M., Fonty, G., and Huis in't Veld, J. H. J. (1999) A computer-controlled system to simulate conditions of the large intestine with peristaltic mixing, water absorption and absorption of fermentation products. *Appl. Microbiol. Biotechnol. 53*, 108–114.

(81) Minekus, M., Marteaul, P., Havenaarl, R., and Huis, J. H. J. (1995) A Multicompartmental Dynamic Computer-controlled Model Simulating the Stomach and Small Intestine. *Altern. to Lab. Anim. 23*, 197–209.

(82) Rajilić-Stojanović, M., Maathuis, A., Heilig, H. G. H. J., Venema, K., de Vos, W. M., and Smidt, H. (2010) Evaluating the microbial diversity of an in vitro model of the human large intestine by phylogenetic microarray analysis. *Microbiology 156*, 3270–81.

(83) Van Nuenen, M. H. M. C., Diederick Meyer, P., and Venema, K. (2003) The effect of various inulins and Clostridium difficile on the metabolic activity of the human colonic microbiota in vitro. *Microb. Ecol. Health Dis. 15*, 137–144.

(84) Venema, K., van Nuenen, M. H. M. C., van den Heuvel, E. G., Pool, W., and van der Vossen, J. M. B. M. (2003) The effect of lactulose on the composition of the intestinal microbiota and short-chain fatty acid production in human volunteers and a computer-controlled model of the proximal large intestine. *Microb. Ecol. Health Dis. 15*, 94–105.

(85) Maathuis, A. J. H., Heuvel, E. G. Van Den, Schoterman, M. H. C., and Venema, K. (2012) Galacto-oligosaccharides have prebiotic activity in a dynamic in vitro colon model using a 13C-labeling technique. *J. Nutr. 142*, 1205–1212.

(86) Egert, M., de Graaf, A. a, Maathuis, A., de Waard, P., Plugge, C. M., Smidt, H., Deutz, N. E. P., Dijkema, C., de Vos, W. M., and Venema, K. (2007) Identification of glucose-fermenting bacteria present in an in vitro model of the human intestine by RNA-stable isotope probing. *FEMS Microbiol. Ecol. 60*, 126–35.

(87) Child, M. W., Kennedy, A., Walker, A. W., Bahrami, B., Macfarlane, S., and Macfarlane, G. T. (2006) Studies on the effect of system retention time on bacterial populations colonizing a three-stage continuous culture model of the human large gut using FISH techniques. *FEMS Microbiol. Ecol. 55*, 299–310.

(88) To, J., Blunt, B., and Chang, D. (2014) Investigating the criteria for accurate quantitative results with Chenomx NMR Suite, pp 1–4.

(89) Ihaka, R., and Gentleman, R. (1996) R: a language for data analysis and graphics. *J. Comput. Graph. Stat. 5*, 299–314.

(90) McDonald, J. A. K. (2013) Development of an in vitro fermentation model to culture the human distal gut microbiota. University of Guelph.

(91) Schroeter, K. F. B. (2014) Adaptation of a Single-Stage Chemostat System to Model Functional Aspects of the Human Distal Gut Microbiota. University of Guelph.

(92) De Boever, P., Deplancke, B., and Verstraete, W. (2000) Fermentation by gut microbiota cultured in a simulator of the human intestinal microbial ecosystem is improved by supplementing a soygerm powder. *J. Nutr. 130*, 2599–606.

(93) Baylis, S. A., Finsterbusch, T., Bannert, N., Blümel, J., and Mankertz, A. (2011) Analysis of porcine circovirus type 1 detected in Rotarix vaccine. *Vaccine 29*, 690–7.

(94) Aranha, H. (2011) Virus safety of biopharmaceuticals: Absence of evidence is not evidence of absence. *Contract Pharma*.

(95) Kerr, A., and Nims, R. (2010) Adventitious viruses detected in biopharmaceutical bulk harvest samples over a 10 Year Period. *PDA J. Pharm. Sci. Technol. 64*, 481–485.

(96) Denyer, S., and Baird, R. (1990) Guide to microbiological control in pharmaceuticals (Rubinstein, M. H., Ed.) 1st ed., pp 158–240. Ellis Horwood Limited, Chichester.

(97) Yaman, A. (2012) Methods of sterilization for controlled release injectable implantable preparations, in *Long Acting Injections and Implants* (Wright, J. C., and Burgess, D. J., Eds.), pp 459–473. Springer US, Boston, MA.

(98) Mosley, G. (2008) Sterility Assurance Level (SAL): The term and its definition continues to cause confusion in the industry. *Pharm. Microbiol. Forum Newsl. 14*, 1–15.

(99) Schleh, M., Romanowski, P., Bhebe, P., Zhang, L., Chinniah, S., Lawrence, B., Bashiri, H., Gaduh, A., Rajurs, V., Rasmussen, B., and Chuck, A. (2009) Susceptibility of Mouse Minute Virus to inactivation by heat in two cell culture media types. *Biotechnol. Prog. 25*, 854–860.

(100) Weaver, B., and Rosenthal, S. (2010) Viral risk mitigation for mammalian cell culture media. *PDA J. Pharm. Sci. Technol. 64*, 436–439.

(101) Pohlscheidt, M., Charaniya, S., Kulenovic, F., Corrales, M., Shiratori, M., Bourret, J., Meier, S., Fallon, E., and Kiss, R. (2014) Implementing high-temperature short-time media treatment in commercial-scale cell culture manufacturing processes. *Appl. Microbiol. Biotechnol. 98*, 2965–2971.

(102) Cao, X., Stimpfl, G., Wen, Z.-Q., Frank, G., and Hunter, G. (2013) Identification and root cause analysis of cell culture media precipitates in the viral deactivation treatment with high-temperature/short-time method. *PDA J. Pharm. Sci. Technol. 67*, 63–73.

(103) Hijnen, W., Beerendonk, E., and Medema, G. (2006) Inactivation credit of UV radiation for viruses, bacteria and protozoan (oo)cysts in water: a review. *Water Res. 40*, 3–22.

(104) Abshire, R. (1988) Ultraviolet radiation: A method of sterilization in the pharmaceutical industry. *Ozone Sci. Eng. 10*, 25–38.

(105) Schmidt, S., and Kauling, J. (2007) Process and laboratory scale UV inactivation of viruses and bacteria using an innovative coiled tube reactor. *Chem. Eng. Technol. 30*, 945–950.

(106) Meng, Q., and Gerba, C. (1996) Comparative inactivation of enteric adenoviruses, poliovirus and coliphages by ultraviolet irradiation. *Water Res. 30*, 2665–2668.

(107) Shanley, J. D. (1982) Ultraviolet irradiation of murine cytomegalovirus. *J. Gen. Virol. 63*, 251–4.

(108) Lytle, C., and Sagripanti, J. (2005) Predicted inactivation of viruses of relevance to biodefense by solar radiation. *J. Virol. 79*, 14244–14252.

(109) Harris, R., Coleman, P., and Morahan, P. (1974) Stability of minute virus of mice to chemical and physical agents. *Appl. Microbiol. 28*, 351–54.

(110) Chevrefils, G., Caron, E., and Wright, H. (2006) UV dose required to achieve incremental log inactivation of bacteria, protozoa and viruses. *IUVA News 8*, 38–45.

(111) Linden, K., Scheible, K., and Posy, P. (2011) Can UV protect the public from Adenovirus in drinking water? *Proc. Water Environ. Fed. 8*, 26–33.

(112) Oms-Oliu, G., Martín-Belloso, O., and Soliva-Fortuny, R. (2008) Pulsed light treatments for food preservation. A review. *Food Bioprocess Technol. 3*, 13–23.

(113) Ansari, I., and Datta, A. (2003) An overview of sterilization methods for packaging materials used in aseptic packaging systems. *Food Bioprod. Process. 81*, 57–65.

(114) Bintsis, T., Litopoulour-Tzanetaki, E., and Robinson, R. K. (2000) Existing and potential applications of ultraviolet light in the food industry–a critical review. *J. Sci. Food Agric. 80*, 637–645.

(115) Wekhof, A., Trompeter, F.-J., and Franken, O. (2001) Pulsed UV Disintegration (PUVD): A new sterilisation mechanism for packaging and broad medical-hospital applications, in *The First International Conference on Ultraviolet Technologies*, pp 1–15.

(116) Yaman, A. (2001) Alternative methods of terminal sterilization for biologically active macromolecules. *Curr. Opin. Drug Discov. Devel. 4*, 760–3.

(117) Agalloco, J., and Akers, J. (2010) Advanced Aseptic Processing Technology (Swarbrick, J., Ed.) 1st ed., pp 278–280. Informa Healthcare, London.

(118) Rubbo, S. D., and Gardner, J. F. (1965) A Review of Sterilization and Disinfection, pp 1–96. Lloyd-Luke (Medical Books) Ltd., Chicago.

(119) Lorenzen, A., James, C. A., and Kennedy, S. W. (1993) Effects of UV irradiation of cell culture medium on PCB-mediated porphyrin accumulation and EROD induction in chick embryo hepatocytes. *Toxicol. In Vitro 7*, 159–66.

(120) Hart, R., and Boychyn, R. M. (2012) Cell Culture media for UVC exposure and methods related thereto. *US Pat. 20,120,214,204*. United States of America.

(121) Qualls, R. G., and Johnson, J. D. (1983) Bioassay and dose measurement in UV disinfection. *Appl. Environ. Microbiol. 45*, 872–877.

(122) Bolton, J. R., and Linden, K. G. (2003) Standardization of methods for fluence (UV dose) determination in bench-scale UV experiments. *J. Environ. Eng. 129*, 209–215.

(123) Pirnie, M., Linden, K. G., and Malley, J. P. J. (2006) Ultraviolet disinfection guidance manual for the final long term 2 enhanced surface water treatment rule. *Environ. Prot.*, pp 1–436. Washington.

(124) Morris, G. A., Barjat, H., and Horne, T. J. (1997) Reference deconvolution methods. *Prog. Nucl. Magn. Reson. Spectrosc. 31*, 197–256.

(125) Adams, D., Korke, R., and Hu, W.-S. (2007) Application of stoichiometric and kinetic analysis to characterize cell growth and product formation, in *Animal Cell Biotechnology Methods and Protocols* (Portner, R., Ed.) 2nd ed., pp 269–284. Humana Press, Totowa, New Jersey.

(126) Vásquez-Vivar, J., Denicola, A. D., Radi, R., and Augusto, O. (1997) Peroxynitrite-mediated decarboxylation of pyruvate to both carbon dioxide and carbon dioxide radical anion. *Chem. Res. Toxicol. 10*, 786–794.

(127) Weil, L., Gordon, W. G., and Buchert, A. R. (1951) Photooxidation of amino acids in the presence of methylene blue. *Arch. Biochem. Biophys. 33*, 90–109.

(128) Handbook, C. R. C. (2014) Vapor Pressure, in *CRC Handbook of Chemistry and Physics* (Haynes, W. M., Ed.) 95th ed., pp 88–117. CRC Press.

(129) Jenner, A. M., Rafter, J., and Halliwell, B. (2005) Human fecal water content of phenolics: the extent of colonic exposure to aromatic compounds. *Free Radic. Biol. Med. 38*, 763–72.

(130) Belenguer, A., Duncan, S. H., Holtrop, G., Anderson, S. E., Lobley, G. E., and Flint, H. J. (2007) Impact of pH on lactate formation and utilization by human fecal microbial communities. *Appl. Environ. Microbiol. 73*, 6526–33.

(131) Walker, A. W., Duncan, S. H., Mcwilliam, E. C., Child, M. W., Flint, H. J., and Leitch, E. C. M. (2005) pH and Peptide Supply Can Radically Alter Bacterial Populations and Short-Chain Fatty Acid Ratios within Microbial Communities from the Human Colon pH and Peptide Supply Can Radically Alter Bacterial Populations and Short-Chain Fatty Acid Ratios within Mi.

(132) Food and Drug Administration. (2001) Guidance for Industry, Bioanalytical Method Validation Guidance for Industry Bioanalytical Method Validation. Rockville, MD.

(133) Elsden, S. R., Hilton, M. G., and Waller, J. M. (1976) The end products of the metabolism of aromatic amino acids by Clostridia. *Arch. Microbiol. 107*, 283–8.

(134) D'Ari, L., and Barker, H. (1985) p-Cresol formation by cell-free extracts of Clostridium difficile. *Arch. Microbiol.* 311–312.

(135) Wishart, D. S., Tzur, D., Knox, C., Eisner, R., Guo, A. C., Young, N., Cheng, D., Jewell, K., Arndt, D., Sawhney, S., Fung, C., Nikolai, L., Lewis, M., Coutouly, M.-A., Forsythe, I., Tang, P., Shrivastava, S., Jeroncic, K., Stothard, P., Amegbey, G., Block, D., Hau, D. D., Wagner, J., Miniaci, J., Clements, M., Gebremedhin, M., Guo, N., Zhang, Y., Duggan, G. E., Macinnis, G. D., Weljie, A. M., Dowlatabadi, R., Bamforth, F., Clive, D., Greiner, R., Li, L., Marrie, T., Sykes, B. D., Vogel, H. J., and Querengesser, L. (2007) HMDB: the Human Metabolome Database. *Nucleic Acids Res. 35*, D521–6.

(136) Bino, R. J., Hall, R. D., Fiehn, O., Kopka, J., Saito, K., Draper, J., Nikolau, B. J., Mendes, P., Roessner-Tunali, U., Beale, M. H., Trethewey, R. N., Lange, B. M., Wurtele, E. S., and Sumner, L. W. (2004) Potential of metabolomics as a functional genomics tool. *Trends Plant Sci. 9*, 418–25.

(137) Tang, J. (2011) Microbial metabolomics. *Curr. Genomics 12*, 391–403.

(138) Martins dos Santos, V. a P., and Damborsky, J. (2010) Systems biology at work. *Curr. Opin. Biotechnol. 21*, 498–501.

(139) Claesson, M. J., Jeffery, I. B., Conde, S., Power, S. E., O'Connor, E. M., Cusack, S., Harris, H. M. B., Coakley, M., Lakshminarayanan, B., O'Sullivan, O., Fitzgerald, G. F., Deane, J., O'Connor, M., Harnedy, N., O'Connor, K., O'Mahony, D., van Sinderen, D., Wallace, M., Brennan, L., Stanton, C., Marchesi, J. R., Fitzgerald, A. P., Shanahan, F., Hill, C., Ross, R. P., and O'Toole, P. W. (2012) Gut microbiota composition correlates with diet and health in the elderly. *Nature 488*, 178–84.

(140) Lenz, E. M., and Wilson, I. D. (2007) Analytical strategies in metabonomics. *J. Proteome Res. 6*, 443–58.

(141) Dettmer, K. (2007) Mass spectrometry-based metabolomics. *Mass Spectrom. Rev. 26*, 51–78.

(142) Ward, J. L., Baker, J. M., and Beale, M. H. (2007) Recent applications of NMR spectroscopy in plant metabolomics. *FEBS J. 274*, 1126–31.

(143) Karunasena, E., McMahon, K. W., Chang, D., and Brashears, M. M. (2014) Host responses to the pathogen Mycobacterium avium subsp. paratuberculosis and beneficial microbes exhibit host sex specificity. *Appl. Environ. Microbiol. 80*, 4481–90.

(144) Cinquin, C., Le Blay, G., Fliss, I., and Lacroix, C. (2004) Immobilization of infant fecal microbiota and utilization in an in vitro colonic fermentation model. *Microb. Ecol. 48*, 128–38.

(145) Team, R. D. C. (2011) R: a language and environment for statistical computing. *R Found. Stat. Comput.*

(146) Wickham, H. (2009) ggplot2: elegant graphics for data analysis (Gentleman, R., Hornik, K., and Parmigiani, G., Eds.), pp 1–202. Springer New York, New York, NY.

(147) Petrof, E. O., Gloor, G. B., Vanner, S. J., Weese, S. J., Carter, D., Daigneault, M. C., Brown, E. M., Schroeter, K., and Allen-Vercoe, E. (2013) Stool substitute transplant therapy for the eradication of Clostridium difficile infection: "RePOOPulating" the gut. *Microbiome 1*, 1–12.

(148) Buckel, W., and Barker, H. a. (1974) Two pathways of glutamate fermentation by anaerobic bacteria. *J. Bacteriol. 117*, 1248–60.

(149) Barker, H. (1981) Amino acid degradation by anaerobic bacteria. *Annu. Rev. Biochem. 50*, 23–40.

(150) Measures, J. (1975) Role of amino acids in osmoregulation of non-halophilic bacteria. *Nature 257*, 398–400.

(151) Giaever, H., and Styrvold, O. (1988) Biochemical and genetic characterization of osmoregulatory trehalose synthesis in Escherichia coli. *J. Bacteriol. 170*, 2841–2849.

(152) Yamamoto, I., Abe, A., Saito, H., and Ishimoto, M. (1984) The pathway of ammonia assimilation in Bacteroides fragilis. *J. Gen. Appl. Microbiol. 30*, 499–508.

(153) Mortensen, P., Clausen, M., Bonnen, H., Hove, H., and Holtug, K. (1992) Colonic fermentation of ispaghula, wheat bran, glucose, and albumin to short-chain fatty acids and ammonia evaluated in vitro in 50 subjects. *J. Parenter. Enter. Nutr. 16*, 433–439.

(154) Macfarlane, G. T., Gibson, G. R., and Cummings, J. H. (1992) Comparison of fermentation reactions in different regions of the human colon. *J. Appl. Bacteriol. 72*, 84–91.

(155) White, D. (2007) The physiology and biochemistry of prokaryotes 3rd ed., pp 388–391. Oxford University Press Inc., New York, NY.

(156) Macy, J., Ljungdahl, L., and Gottschalk, G. (1978) Pathway of succinate and propionate formation in Bacteroides fragilis. *J. Bacteriol. 134*, 84–91.

(157) Burlingame, R., and Chapman, P. J. (1983) Catabolism of phenylpropionic acid and its 3-hydroxy derivative by Escherichia coli. *J. Bacteriol. 155*, 113–21.

(158) Díaz, E., Ferrández, a, and García, J. L. (1998) Characterization of the hca cluster encoding the dioxygenolytic pathway for initial catabolism of 3-phenylpropionic acid in Escherichia coli K-12. *J. Bacteriol. 180*, 2915–23.

(159) Moss, C., Lambert, M., and Goldsmith, D. (1970) Production of hydrocinnamic acid by clostridia. *Appl. Microbiol. 19*, 375–379.

(160) Ploux, O., Soularue, P., Marquet, A., Gloeckler, R., and Lemoine, Y. (1992) Investigation of the first step of biotin biosynthesis in Bacillus sphaericus. Purification and characterization of the pimeloyl-CoA synthase, and uptake of pimelate. *Biochem. J. 287*, 685–90.

(161) Rosner, A. (1975) Control of lysine biosynthesis in Bacillus subtilis: inhibition of diaminopimelate decarboxylase by lysine. *J. Bacteriol. 121*, 20–28.

(162) Chapman, P., and Duggleby, R. (1967) Dicarboxylic acid catabolism by bacteria. *Biochem. J 103*, 7–9.

(163) Mart, N., Gibello, A., and Ndez, F. (1991) Catabolism of 3- and 4-hydroxyphenylacetic acid by Kfebsieffa pneumoniae 621–628.

(164) Mohamed, M. E., Seyfried, B., Tschech, A., and Fuchs, G. (1993) Anaerobic oxidation of phenylacetate and 4-hyrdoxyphenylacetate to benzoyl-coenzyme A and CO2 in denytrifying Pseudomonas sp. *Arch. Microbiol. 159*, 563–573.

(165) Gibson, G. R., Cummings, J. H., and Macfarlane, G. T. (1991) Growth and activities of sulphate-reducing bacteria in gut contents of healthy subjects and patients with ulcerative colitis. *FEMS Microbiol. Lett. 86*, 103–112.

(166) Smith, E., and Macfarlane, G. (1998) Enumeration of amino acid fermenting bacteria in the human large intestine: effects of pH and starch on peptide metabolism and dissimilation of amino acids. *FEMS Microbiol. Ecol. 25*, 355–368.

(167) Hilton, M. G., Mead, G. C., and Elsden, S. R. (1975) The metabolism of pyrimidines by proteolytic clostridia. *Arch. Microbiol. 102*, 145–9.

(168) Goldfine, H., and Stadtman, E. R. (1960) Propionic acid metabolism: V. The conversion of β-alanine to propionic acid by cell-free extracts of Clostridium propionicum. *J. Biol. Chem. 235*, 2238–2245.

(169) Hayaishi, O., Nishizuka, Y., Tatibana, M., Takeshita, M., and Kuno, S. (1961) Enzymatic studies on the metabolism of β-alanine. *J. Biol. Chem. 236*, 781–790.

(170) Newmark, H. L., and Lupton, J. R. (1990) Determinants and consequences of colonic luminal pH : Implications for colon cancer. *Nutr. Cancer 14*, 161–173.

(171) Schink, B. (1984) Fermentation of tartrate enantiomers by anaerobic bacteria, and description of two new species of strict anaerobes, Ruminococcus pasteurii and Ilyobacter tartaricus. *Arch. Microbiol. 139*, 409–414.

(172) Chadwick, V., Vince, A., Killingley, M., and Wrong, O. (1978) The metabolism of tartrate in man and the rat. *Clin Sci Mol Med 54*, 273–281.

(173) Hinton, A. J., and Hume, M. (1997) Inhibition of Listeria monocytogenes growth by Veillonella cultured on tartrate medium. *Clin. Infect. Dis. 25*, S120.

(174) La Rivière, J. (1958) On the microbial metabolism of the tartaric acid isomers. Uitgeverij Waltman: Delft.

(175) Deppenmeier, U., Hoffmeister, M., and Prust, C. (2002) Biochemistry and biotechnological applications of Gluconobacter strains. *Appl. Microbiol. Biotechnol. 60*, 233–42.

(176) Chandrashekar, K., Felse, P., and Panda, T. (1999) Optimization of temperature and initial pH and kinetic analysis of tartaric acid production by Gluconobacter suboxydans. *Bioprocess Eng. 20*, 1–5.

(177) Vogels, G. D., and Van der Drift, C. (1976) Degradation of purines and pyrimidines by microorganisms. *Bacteriol. Rev. 40*, 403–68.

(178) Wolpert, E., Phillips, S., and Summerskill, W. (1971) Transport of urea and ammonia production in the human colon. *Lancet* 1387–1390.

(179) Dai, Z.-L., Li, X.-L., Xi, P.-B., Zhang, J., Wu, G., and Zhu, W.-Y. (2013) L-Glutamine regulates amino acid utilization by intestinal bacteria. *Amino Acids 45*, 501–12.

(180) Kristoffersen, T., and Nelson, F. E. (1955) Degradation of amino acids by Lactobacillus casei and some factors influencing deamination of serine. *Appl. Microbiol. 3*, 268–73.

(181) Hayward, H. R., and Stadtman, T. C. (1959) Anarobic degradation of choline I., Vibrio cholinicus n. sp: Fermentation of Choline by an anaerobic, cytochrome-producing bacterium *78*, 557–561.

(182) Neill, A. R., Grime, D. W., and Dawson, R. M. (1978) Conversion of choline methyl groups through trimethylamine into methane in the rumen. *Biochem. J. 170*, 529–35.

(183) Craciun, S., and Balskus, E. P. (2012) Microbial conversion of choline to trimethylamine requires a glycyl radical enzyme. *Proc. Natl. Acad. Sci. U. S. A. 109*, 21307–12.

(184) Fink, R., McGaughey, C., Cline, R., and Fink, K. (1956) Metabolism of intermediate pyrimidine reduction products in vitro. *J. Biol. Chem. 218*, 1–7.

(185) Lara, F. (1952) On the decomposition of pyrimidines by bacteria I.: Studies by means of the technique of simultaneous adaptation. *J. Bacteriol. 64*, 271–277.

(186) Friedrich, M., and Schink, B. (1993) Hydrogen formation from glycolate driven by reversed electron transport in membrane vesicles of a syntrophic glycolate-oxidizing bacterium. *Eur. J. Biochem. 217*, 233–40.

(187) Savage, G. M. (1974) Lincomycin and clindamycin: their role in chemotherapy of anaerobic and microaerophilic infections. *Postepy Hig. Med. Dosw. 28*, 573–85.

(188) Chang, F., Sih, C., and Weisblum, B. (1966) Lincomycin, an inhibitor of aminoacyl sRNA binding to ribosomes. *Proc. Natl. Acad. Sci. U. S. A. 55*, 431–438.

(189) Kasten, M. J. (1999) Clindamycin, metronidazole, and chloramphenicol. *Mayo Clin. Proc. 74*, 825–33.

(190) Heimdahl, A., and Nord, C. (1982) Effect of erythromycin and clindamycin on the indigenous human anaerobic flora and new colonization of the gastrointestinal tract. *Eur. J. Clin. Microbiol. 1*, 38–48.

(191) Centers for Disease Control and Prevention. (1995) Recommendations for preventing the spread of vancomycin resistance: recommendations of the Hospital Infection Control Practices Advisory Committee (HICPAC). *Morbitity Mortal. Wkly. Rep. 44*, 105–113.

(192) Loll, P. J., Kaplan, J., Selinsky, B. S., and Axelsen, P. H. (1999) Vancomycin binding to low-affinity ligands: delineating a minimum set of interactions necessary for high-affinity binding. *J. Med. Chem. 42*, 4714–9.

(193) Walsh, C. T. (1993) Vancomycin resistance: decoding the molecular logic. *Science 261*, 308–9.

(194) Guyton, A. C., and Hall, J. E. I. (2006) Textbook of medical psychology 11th ed., pp 748–760. Elsevier Inc., Philadelphia.

(195) Everly, G. S., and Lating, J. M. (2013) The anatomy and physiology of the human stress response, in *A clinical guide to the treatment of the human stress response* (Everly, G. S. J., and Lating, J. M., Eds.) 3rd ed., pp 15–48. Springer New York, New York, NY.

(196) Eisenhofer, G., Aneman, A., Hooper, D., Rundqvist, B., and Friberg, P. (1996) Mesenteric organ production, hepatic metabolism, and renal elimination of norepinephrine and its metabolites in humans. *J. Neurochem. 66*, 1565–73.

(197) Hawrelak, J. a, and Myers, S. P. (2004) The causes of intestinal dysbiosis: a review. *Altern. Med. Rev. 9*, 180–97.

(198) Bailey, M., and Coe, C. (1999) Maternal separation disrupts the integrity of the intestinal microflora in infant rhesus monkeys. *Dev. Psychobiol. 35*, 146–155.

(199) O'Donnell, P. M., Aviles, H., Lyte, M., and Sonnenfeld, G. (2006) Enhancement of in vitro growth of pathogenic bacteria by norepinephrine: importance of inoculum density and role of transferrin. *Appl. Environ. Microbiol. 72*, 5097–9.

(200) Bailey, M. T. (2010) Psychological stress, immunity, and the effects on indigenous microflora, in *Microbial Endocrinology* (Lyte, M., and Freestone, P. P. E., Eds.), pp 191–212. Springer New York, New York, NY.

(201) Lyte, M., and Ernst, S. (1992) Catecholamine induced growth of gram negative bacteria. *Life Sci. 50*, 203–212.

(202) Yamamoto, I., Saito, H., and Ishimoto, M. (1987) Regulation of synthesis and reversible inactivation in vivo of dual coenzyme-specific glutamate dehydrogenase in Bacteroides fragilis. *J. Gen. Microbiol. 133*, 2773–80.

(203) Gorbach, S. L., Spanknebel, G., Weinstein, L., Plaut, A. G., Nahas, L., and Levitan, R. (1969) Studies of intestinal microflora. VIII. Effect of lincomycin on the microbial population of the human intestine. *J. Infect. Dis. 120*, 298–304.

(204) Edwards, C., Duerden, B., and Read, N. (1986) Effect of clindamycin on the ability of a continuous culture of colonic bacteria to ferment carbohydrate. *Gut 27*, 411–417.

(205) Thauer, R. K., Jungermann, K., and Decker, K. (1977) Energy conservation in chemotrophic anaerobic bacteria. *Bacteriol. Rev. 41*, 809.

(206) Lowe, S. E., Jain, M. K., and Zeikus, J. G. (1993) Biology, ecology, and biotechnological applications of anaerobic bacteria adapted to environmental stresses in temperature, pH, salinity, or substrates. *Microbiol. Rev. 57*, 451–509.

(207) Hold, G. L., Schwiertz, A., Aminov, R., Blaut, M., and Flint, H. J. (2003) Oligonucleotide probes that detect quantitatively significant groups of butyrate-producing bacteria in human feces. *Appl. Environ. Microbiol. 69*, 4320–4324.

(208) Yap, I. K., Li, J. V, Saric, J., Martin, F.-P., Davies, H., Yulan, W., D, W. I., Nicholson, J. K., Utzinger, J., Marchesi, J. R., and Holmes, E. (2008) Metabonomic and microbiological analysis of the dynamic effect of vancomycin-induced gut microbiota modification in the mouse. *J. Proteome Res. 7*, 3718–3728.

(209) Woodmansey, E. (2004) Comparison of compositions and metabolic activities of fecal microbiotas in young adults and in antibiotic-treated and non-antibiotic-treated elderly subjects. *Appl. Environ. Microbiol. 70*, 6113–6122.

(210) Zeikus, J. G. (1980) Chemical and fuel production by anaerobic bacteria. *Annu. Rev. Microbiol. 34*, 423–64.

(211) Meiberg, J. B. M., and Harder, W. (1978) Aerobic and anaerobic metabolism of trimethylamine, dimethylamine and methylamine in Hyphomicrobium X. *J. Gen. Microbiol. 106*, 265–276.

(212) Patterson, J., and Hespell, R. (1979) Trimethylamine and methylamine as growth substrates for rumen bacteria andMethanosarcina barkeri. *Curr. Microbiol. 3*, 79–83.

(213) Ban, J., Vitale, L., and Kos, E. (1972) Thymine and uracil catabolism in Escherichia coli. *J. Gen. Microbiol. 73*, 267–72.

(214) Gibbons, R. J., and Macdonald, J. B. (1960) Hemin and vitamin K compounds as required factors for the cultivation of certain strains of Bacteroides melaninogenicus. *J. Bacteriol. 80*, 164–70.

(215) Stojiljkovic, I., and Hantke, K. (1992) Hemin uptake system of Yersinia enterocolitica: similarities with other TonB-dependent systems in gram-negative bacteria. *EMBO J. 11*, 4359–4367.

(216) Spízek, J., and Rezanka, T. (2004) Lincomycin, clindamycin and their applications. *Appl. Microbiol. Biotechnol. 64*, 455–64.

Appendix A Supplementary Figures and Tables

A.1 Materials and Methods

Table A1. Cultured bacterial isolates for MET-1.[6]

Higher taxonomic group		Closest species match	%	Relative abundance (by biomass)
Actinobacteria		Bifidobacterium adolescentis (two different strains)	99.79 99.79	1.5 1.5
		Bifidobacterium longum (two different strains)	99.86 99.16	0.5 2
		Collinsella aerofaciens	98.73	1
		Bacteroides ovatus	99.52	1.5
		Parabacteroides distasonis	99.45	1.5
Firmicutes	Bacilli	Lactobacillus casei/paracasei	99.47	1
		Lactobacillus casei	99.74	1
		Streptococcus mitis	99.79	0.75
	Clostridium cluster IV	Eubacterium desmolans	94.9	1
		Faecalibacterium prausnitzii	99.17	2
	Clostridium cluster VIII	Clostridium cocleatum	91.92	1
	Clostridium cluster IX	Acidaminococcus intestini	100	1
	Clostridium cluster XIVa	Blautia sp.	99.55	0.75
		Dorea longicatena	99.62	1
		(two different strains)	99.6	2
		Eubacterium eligens	98.15	2
		Eubacterium rectale (four different strains)	99.59 99.6 99.19 99.53	2 1.5 1 1
		Eubacterium ventriosum	100	1
		Lachnospira pectinoshiza	95.22	0.5
		Roseburia faecalis	99.65	1.5
		Roseburia intestinalis	100	1

[6]Adapted from Table 2.3 of Development of an In Vitro Fermentation Model to Culture the Human Distal Gut Microbiota[90]. The species column refers to the closest species match, inferred by alignment of 16S rRNA sequence to GreenGenes database. The "%" column refers to the percent identity closest match. Units are presented as the equivalent of 1 X 10 µL loopful of bacterial culture scraped from petri dish plates.

		Ruminococcus torques	99.15	1
		(two different strains)	99.29	2
		Ruminococcus sp. (two different strains)	97.35 98.4	1.5 0.5
	Clostridium cluster XV	Eubacterium limosum	97.05	1.5
Proteobacteria		Escherichia coli (two different strains)	99.8 99.6	0.5 -
		Raoultella sp.	99.4	1

Table A2. Cultured bacterial isolates for MET-2.[7]

Higher taxonomic group	Closest species match	%	Relative abundance (by biomass)
Actinobacteria	*Bifidobacterium longum*	100	1
	Collinsella aerofaciens	100	0.5
	Microbacterium schleiferi	99.34	1.5
	Aldercreutzia equolifaciens	99.76	1
	Micrococcus luteus	97.04	1
Bacteroidetes	*Bacteroides ovatus*	100	0.5
	Parabacteroides merdae	100	1
Firmicutes	*Eubacterium rectale*	100	1
	Blautia luti	98.91	1
	Roseburia hominis	99.04	1
	Roseburia lactaris	95.07	0.5
	Ruminococcus albus	96.96	1
	Eubacterium eligens	96.78	1
	Ruminococcus torques (two different strains)	99.27 100	1 1
	Roseburia intestinalis	100	1
	Eubacterium fissicatena	97.67	1
	Eubacterium ventriosum	97.37	1
	Blautia coccoides	99.85	0.25
	Blautia hydrogenotrophica	100	0.1
	Dorea longicatena	100	0.25
	Dorea formicigenerans	99.49	0.25
	Clostridium ramosum	96.14	0.25
	Eubacterium limosum	99.25	0.75
	Streptococcus thermophilus	100	0.25
	Bacillus simplex	98.7	0.25
	Coprococcus catus	99.19	0.25
	Flavonifractor plautii	96.21	0.25

[7]The species column refers to the closest species match, inferred by alignment of 16S rRNA sequence to GreenGenes database. The "%" column refers to the percent identity closest match. Units are presented as the equivalent of 1 X 10 µL loopful of bacterial culture scraped from petri dish plates.

	Streptococcus mitis	100	0.3
	Phascolarctobacterium sp.	99.85	0.1
Proteobacteria	*Escherichia coli*	100	0.25
	Parasutterella excrementihominis	100	0.5

Table A3. Cultured bacterial isolates for MET-2A, MET-2B, MET-3A, and MET-3B.[8]

Higher taxonomic group	Closest species match	%	Relative abundance (by biomass)							
			Donor D				Donor A			
			Run 30		Run 33		Run 31		Run 32	
			MET-2A	MET-2B	MET-2A*	MET-2B*	MET-3A	MET-3B	MET-3A*	MET-3B*
Actinobacteria	*Bifidobacterium longum* (three different strains)	99.86	1	0.50	1	0.50	-	-	-	-
		99.16	1.5	0.75	1.5	0.75	-	-	-	-
		100.00	-	-	-	-	1	0.50	1	0.50
	Bifidobacterium adolescentis (two different strain)‡	99.79	1	1	0.25	0.25	-	-	-	-
		99.79	1	1	1	1	-	-	-	-
	Collinsella aerofaciens (two different strains)	98.73	0.50	1	0.50	1	-	-	-	-
		100.00	-	-	-	-	0.50	1	0.50	1
	Microbacterium schleiferi	99.34	-	-	-	-	1.5	0.75	1.5	0.75
	Aldercreutzia equolfaciens‡	99.76	-	-	-	-	1	1	0.50	0.50
	Micrococcus luteus	97.04	-	-	-	-	1	1	1	1
Bacteroidetes	*Bacteroides ovatus* (two different strains)	99.52	0.50	1.5	0.50	1.5	-	-	-	-
		100.00	-	-	-	-	0.50	1.5	0.50	1.5
	Parabacteroides distasonis	99.45	1	1	1	1	-	-	-	-
	Parabacteroides merdae	100.00	-	-	-	-	1	1	1	1
Firmicutes	*Eubacterium rectale* (five different strains)	99.59	1	0.25	1	0.25	-	-	-	-
		99.60	1	0.25	1	0.25	-	-	-	-
		99.19	1	1	1	1	-	-	-	-
		99.53	0.50	1	0.50	1	-	-	-	-
		100.00	-	-	-	-	1	0.25	1	0.25
	Blautia luti	98.91	-	-	-	-	1	0.25	1	0.25
	Roseburia hominis	99.04	-	-	-	-	1	1	1	1
	Roseburia lactaris	95.07	-	-	-	-	0.50	1	0.50	1
	Faecalibacterium prausnitzii	99.17	1	0.25	1	0.25	-	-	-	-

[8]Adapted from Table 2.3 of Adaptation of a Single-Stage Chemostat System to Model Functional Aspects of the Human Distal Gut Microbiota[91]. The species column refers to the closest species match, inferred by alignment of 16S rRNA sequence to GreenGenes database. The "%" column refers to the percent identity closest match. Units are presented as the equivalent of 1 X 10 µL loopful of bacterial culture scraped from petri dish plates. ‡ Bacterial isolates that differed between inocula from replicate runs. *Has slight deviation in bacterial composition compared to original defined communities in Run 30 and Run 31

	Ruminococcus albus	96.96	-	-	-	-	1	0.25	1	0.25
Firmicutes	*Eubacterium eligens* (2 different strains)	98.15 96.78	1 -	0.25 -	1 -	0.25 -	- 1	- 0.25	- 1	- 0.25
	Ruminococcus torques (four different strains)	99.15 99.29 99.27 100.00	1.5 1.5 - -	0.50 0.75 - -	1.5 1.5 - -	0.50 0.75 - -	- - 1.5 1.5	- - 0.50 0.75	- - 1.5 1.5	- - 0.50 0.75
	Roseburia intestinalis (two different strains)	100.00 100.00	1 -	1 -	1 -	1 -	- 1	- 1	- 1	- 1
	Roseburia faecalis	99.65	1	1	1	1	-	-	-	-
	Eubacterium fissicatena	97.67	-	-	-	-	1	1	1	1
	Eubacterium ventriosum (two different strains)[‡]	100.00 97.37	1 -	1 -	0.10 -	0.10 -	- 1	- 1	- 0.25	- 0.25
	Ruminococcus obeum (two different strains)	99.55 97.35	0.50 0.25	1.5 1	0.50 0.25	1.5 1	- -	- -	- -	- -
	Ruminococcus producta	98.40	0.25	0.50	0.25	0.50	-	-	-	-
	Blautia coccoides	99.85	-	-	-	-	0.25	0.50	0.25	0.50
	Blautia hydrogenotrophica	100.00	-	-	-	-	0.10	0.50	0.10	0.50
	Dorea longicatena (three different strains)	99.62 99.60 100.00	0.25 0.25 -	1.5 1 -	0.25 0.25 -	1.5 1 -	- - 0.25	- - 1	- - 0.25	- - 1
	Dorea formicigenerans	99.49	-	-	-	-	0.25	1.5	0.25	1.5
	Clostridium cocleatum[‡]	91.92	0.25	1	0.25	0.50	-	-	-	-
	Clostridium ramosum	96.14	-	-	-	-	0.25	1	0.25	1
	Eubacterium limosum (two different strains)	97.05 99.25	0.75 -	1.5 -	0.75 -	1.5 -	- 0.75	- 1.5	- 0.75	- 1.5
	Lactobacillus casei	99.74	0.25	1	0.25	1	-	-	-	-
	Streptococcus thermophilus	100.00	-	-	-	-	0.25	1	0.25	1
	Lactobacillus paracasei	99.47	0.25	1	0.25	1	-	-	-	-
	Bacillus simplex	98.70	-	-	-	-	0.25	1	0.25	1
Firmicutes	*Coprococcus catus*	99.19	-	-	-	-	0.25	1	0.25	1
	Eubacterium	94.90	0.25	1	0.25	1	-	-	-	-

	desmolans									
	Flavonifractor plautii	96.21	-	-	-	-	0.25	1	0.25	1
	Streptococcus mitis (two different strains)	99.79 100.00	0.30 -	0.30 -	0.30 -	0.30 -	- 0.30	- 0.30	- 0.30	- 0.30
	Acidaminococcus intestini	100.00	1.5	0.75	1.5	0.75	-	-	-	-
	Phascolarctobacterium sp.	99.85	-	-	-	-	0.10	0.05	0.10	0.05
Proteobacteria	*Escherichia coli* (two different strains)	99.80 100.00	0.25 -	0.50 -	0.25 -	0.50 -	- 0.25	- 0.50	- 0.25	- 0.50
	Raoultella sp.	99.40	0.50	0.25	0.50	0.25	-	-	-	-
	Parasutterella excrementihominis	100.00	-	-	-	-	0.5	0.25	0.5	0.25

A.2 Standardization of Fecal Water Samples and Analysis

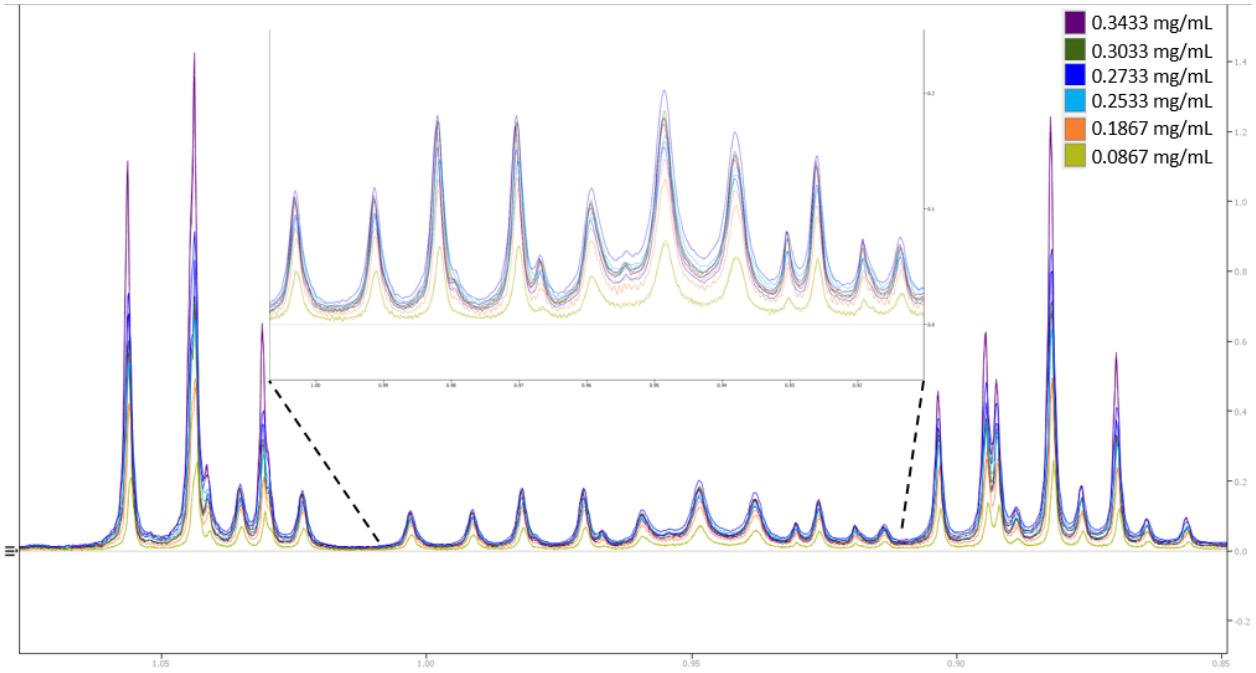

Figure A1. NMR Spectra of fecal water at various concentrations (0.85-1.05 ppm). Includes the NMR signature or partial signature of isobutyrate, propionate, isoleucine, valine, leucine, 2-oxoisocaproate, isovalerate, and butyrate from left to right. Inset is zoomed view of 0.90-1.01 ppm, includes NMR signatures or partial signatures of isoleucine, valine, leucine and 2-oxoisocaproate from left to right.

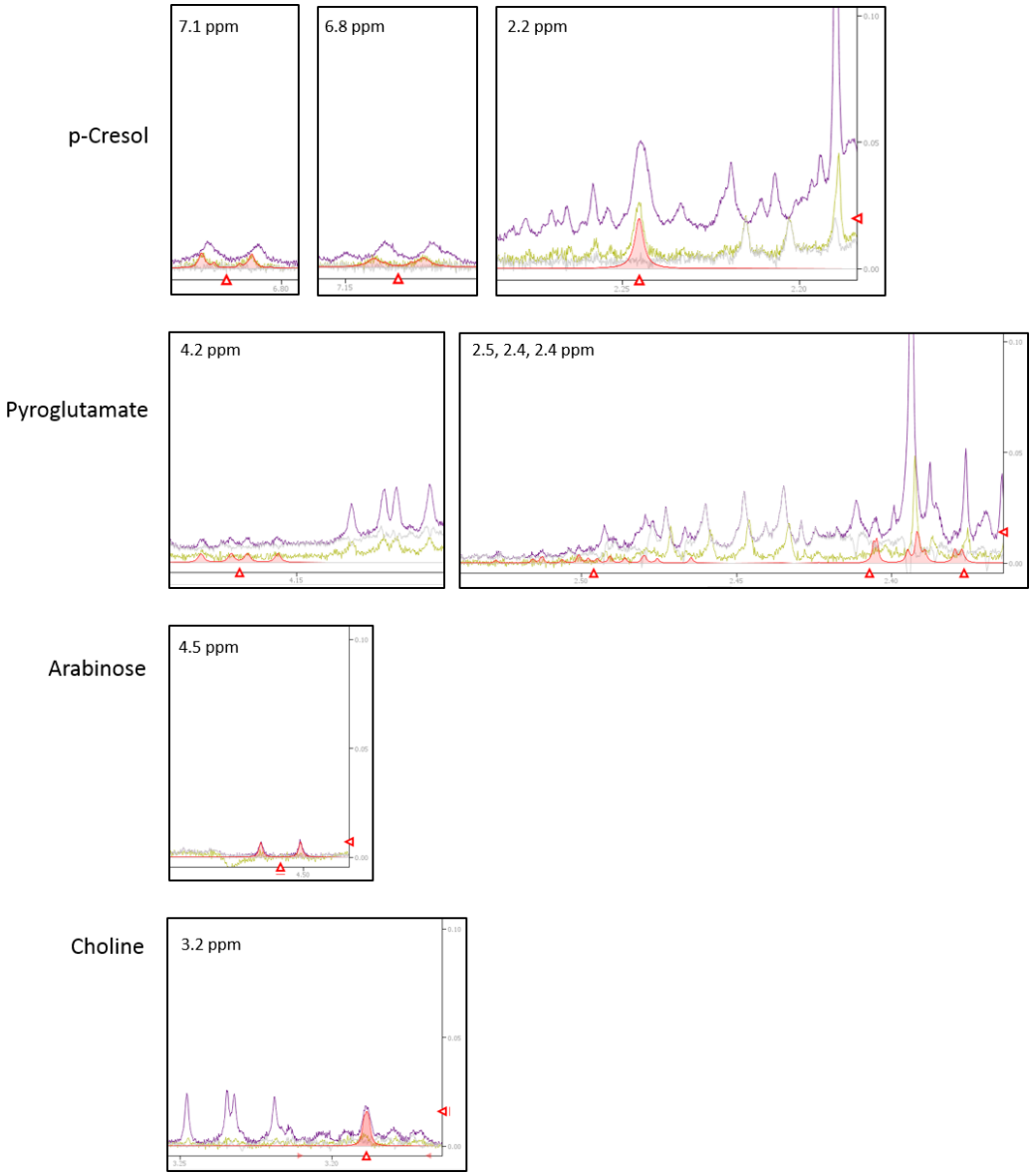

Figure A2. Profiling confidence impacted for certain compounds. Purple represents spectrum of 34.33% fecal water, and light green represents spectrum of 8.670% fecal water. For p-cresol, grey line represents the subtraction line between the 8.670% spectrum and area of profiled compounds in that spectrum. The p-Cresol signal, highlighted in red, represents the profiling of that compound fitted to the 8.670% sample spectrum. For all other compounds, the grey line represents the subtraction line between 34.33% fecal water and the area of all compounds profiled in that spectrum. The compound signals for pyroglutamate, arabinose and choline, highlighted in red, represent the profiling assignment of those compounds fitted to the 34.33% sample spectrum. Reason for demonstrating pyroglutamate, arabinose and choline relative to the high-concentration spectrum is because those compounds are not easily visualized in the low-concentration sample spectrum, due to low signal-to-noise ratio. Note: only compound clusters that were used in facilitating compound identification and quantifications are depicted.

A.3 Sample Preparation Procedure for Metabolomic Analysis

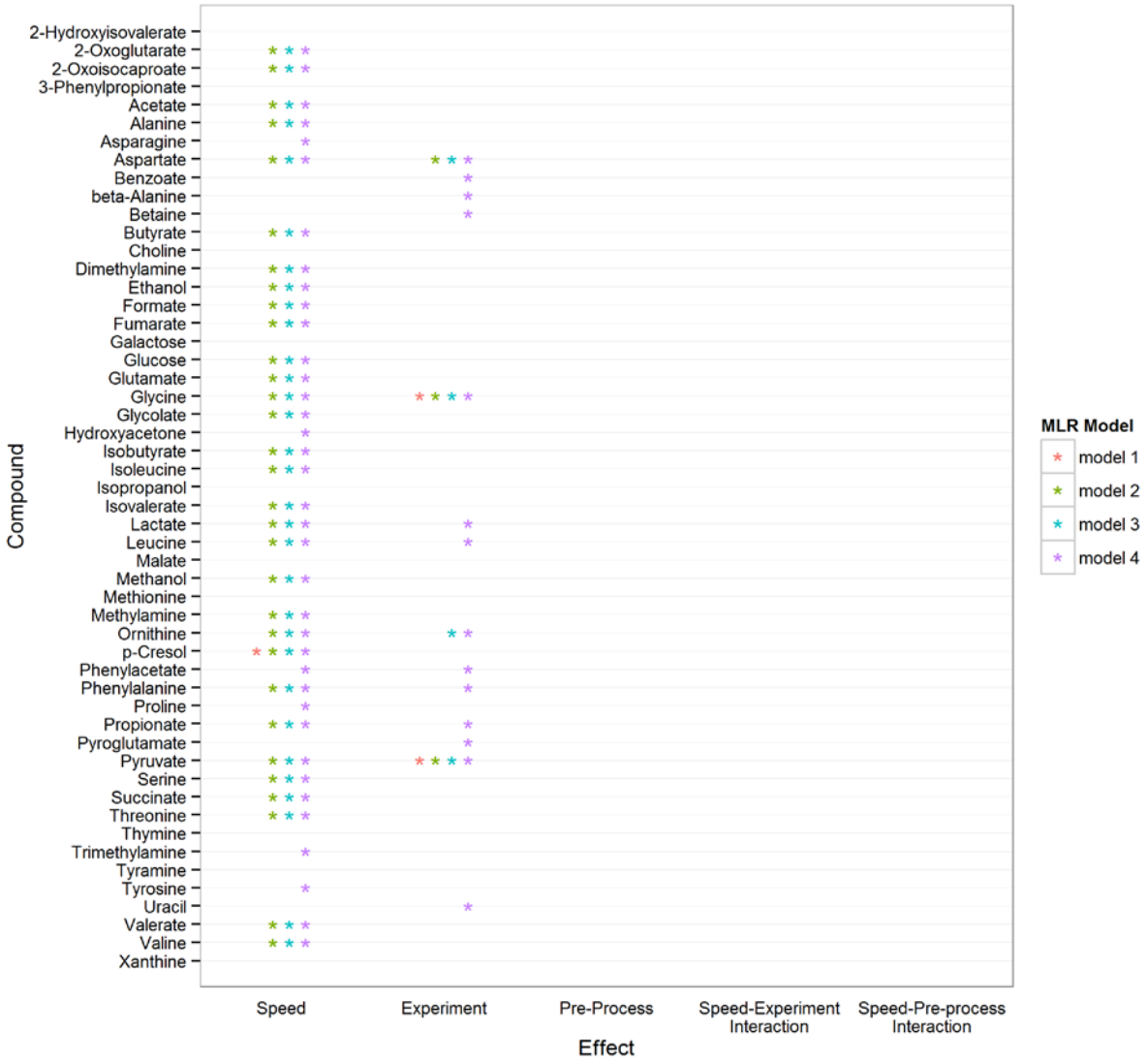

Figure A3. Significance of terms in each iteration of model simplification, with model 1 being the most complex model, and model 4 being the simplest model. Significant terms indicated with a star (*).

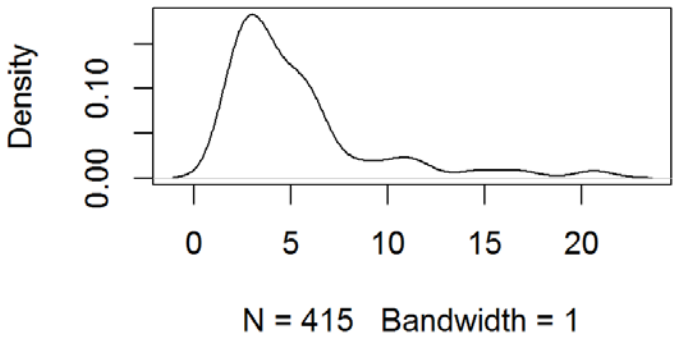

Figure A4. Evaluation of profiling precision based on relative standard deviation of compound concentrations observed in single-step filtered samples from Experiment 2.

Table A4. Compounds that did not demonstrate significant linear trend with changes in centrifugal force, based on simplest MLR model.

Compound	Slope	Standard error	T statistic	P value
3-Phenylpropionate	-9.61E-08	5.07E-08	-1.89346	0.064462
beta-Alanine	-4.98E-08	4.71E-08	-1.05815	0.295394
Choline	8.49E-10	3.59E-09	0.236667	0.813943
Pyroglutamate	-2.38E-07	1.48E-07	-1.61043	0.114001
Tyramine	2.39E-08	3.26E-08	0.734012	0.466588
Xanthine	-3.17E-08	3.69E-08	-0.85923	0.394576

Table A5. Compounds that demonstrated the same trends in concentration change with ultracentrifuge speed in both Experiment 1 and Experiment 2.

Compound	Experiment Bias	Standard Error	T Statistic	P-value
2-Hydroxyisovalerate	-0.00165	0.000561	-2.94227	0.005046
2-Oxoglutarate	-0.00253	0.001059	-2.38909	0.020956
2-Oxoisocaproate	-0.00024	0.000843	-0.28491	0.776966
3-Phenylpropionate	-0.00038	0.001903	-0.19831	0.843658
Acetate	2.170932	0.753433	2.881389	0.005949
Alanine	-0.01835	0.016419	-1.11733	0.269532
Asparagine	-0.00698	0.00222	-3.14535	0.002875
Butyrate	0.197922	0.089398	2.213936	0.031721
Choline	0.000274	0.000135	2.038151	0.047184
Dimethylamine	0.001538	0.000924	1.663707	0.102826
Ethanol	0.258782	0.102993	2.512618	0.015474
Formate	0.001515	0.002143	0.707101	0.483071
Fumarate	-6.57E-05	0.000249	-0.26356	0.793271
Galactose	-0.00855	0.004827	-1.77141	0.082977
Glucose	-0.00346	0.004161	-0.83057	0.410415
Glutamate	0.013564	0.014769	0.918433	0.363082
Glycolate	0.001112	0.002222	0.500279	0.619213
Hydroxyacetone	-0.00176	0.000749	-2.35505	0.022747
Isobutyrate	0.028285	0.016396	1.725171	0.091067
Isoleucine	-0.00475	0.010084	-0.47081	0.639952
Isopropanol	0.001274	0.002859	0.445497	0.658006
Isovalerate	-0.00559	0.008376	-0.66739	0.507788
Malate	0.002312	0.002725	0.848437	0.400495
Methanol	0.005318	0.002503	2.124866	0.038885
Methionine	0.001398	0.002021	0.691921	0.492391
Methylamine	0.006067	0.002394	2.53447	0.014652
p-Cresol	0.008506	0.003623	2.347627	0.023155
Proline	0.012527	0.003986	3.142632	0.002898
Serine	0.005399	0.003212	1.681038	0.09939
Succinate	0.065393	0.019404	3.37	0.00151
Threonine	6.22E-05	0.003425	0.018166	0.985584
Thymine	0.000269	0.00145	0.185337	0.853762
Trimethylamine	3.07E-05	0.000618	0.049737	0.960542
Tyramine	-0.00107	0.001222	-0.87233	0.387465
Tyrosine	0.013931	0.004123	3.378554	0.001473
Valerate	0.007871	0.028608	0.275138	0.784416

Valine	-0.02753	0.013235	-2.07976	0.043027
Xanthine	0.003677	0.001385	2.655878	0.010767

A.4 Effects of Varying Inoculant Biomass Proportions

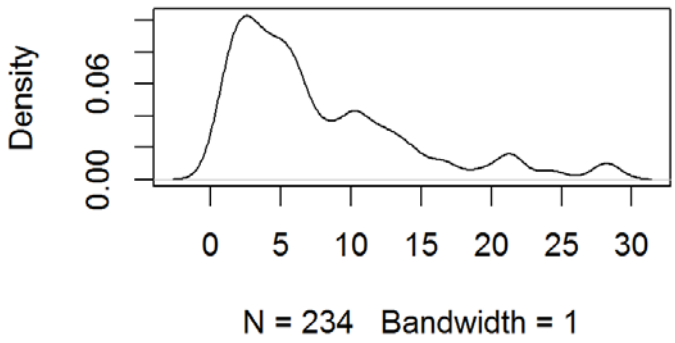

Figure A5. Density plot of the relative standard deviation of triplicate samples.

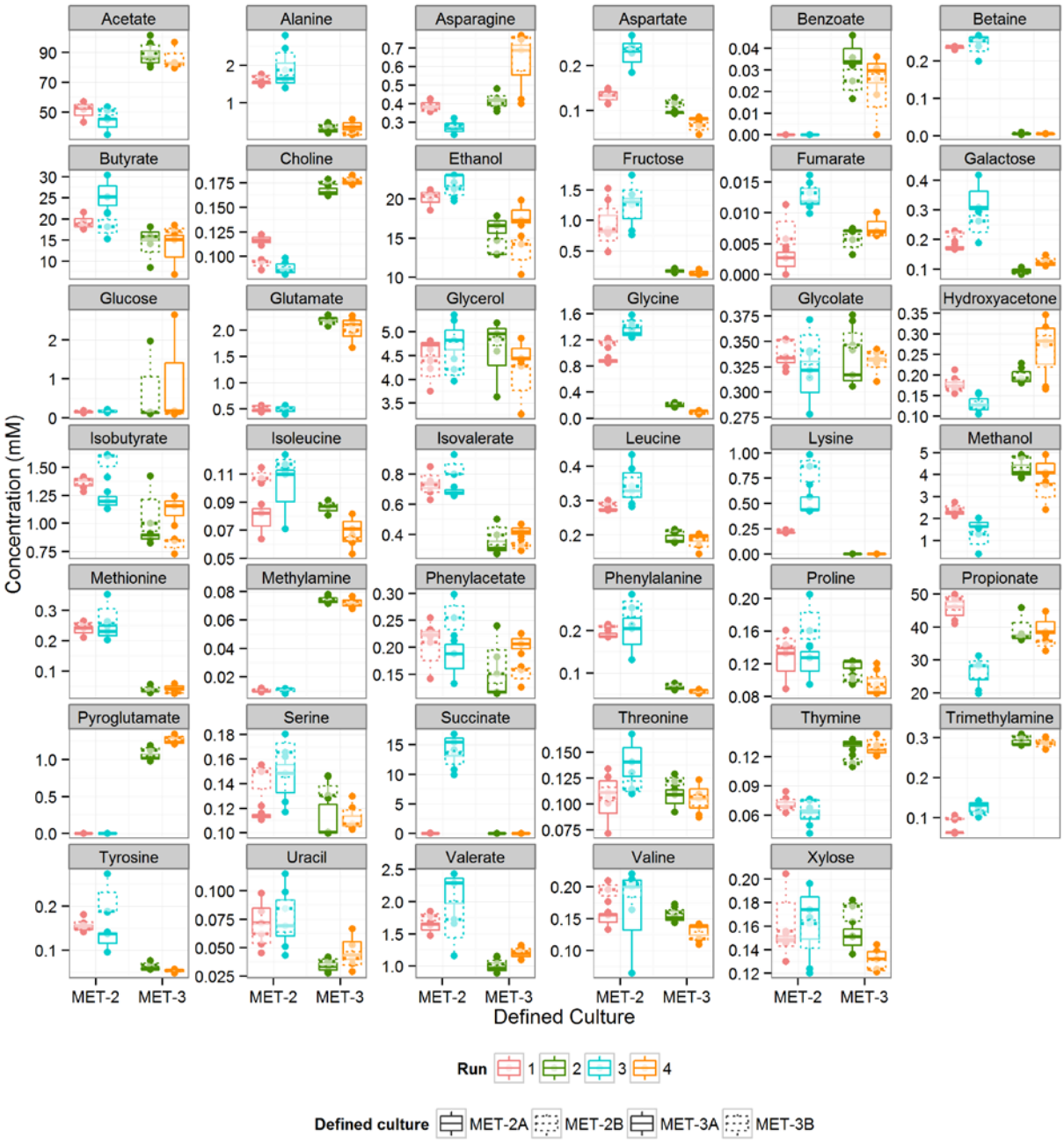

Figure A6. Profile overview of MET-2A and MET-3A and their reciprocal cultures, MET-2B and MET-3B, respectively.

A.5 METABOLOMIC ANALYSIS OF THE HUMAN GUT MICROBIOTA: COMPARISON OF FECES-DERIVED COMMUNITIES AND DEFINED MIXED COMMUNITIES

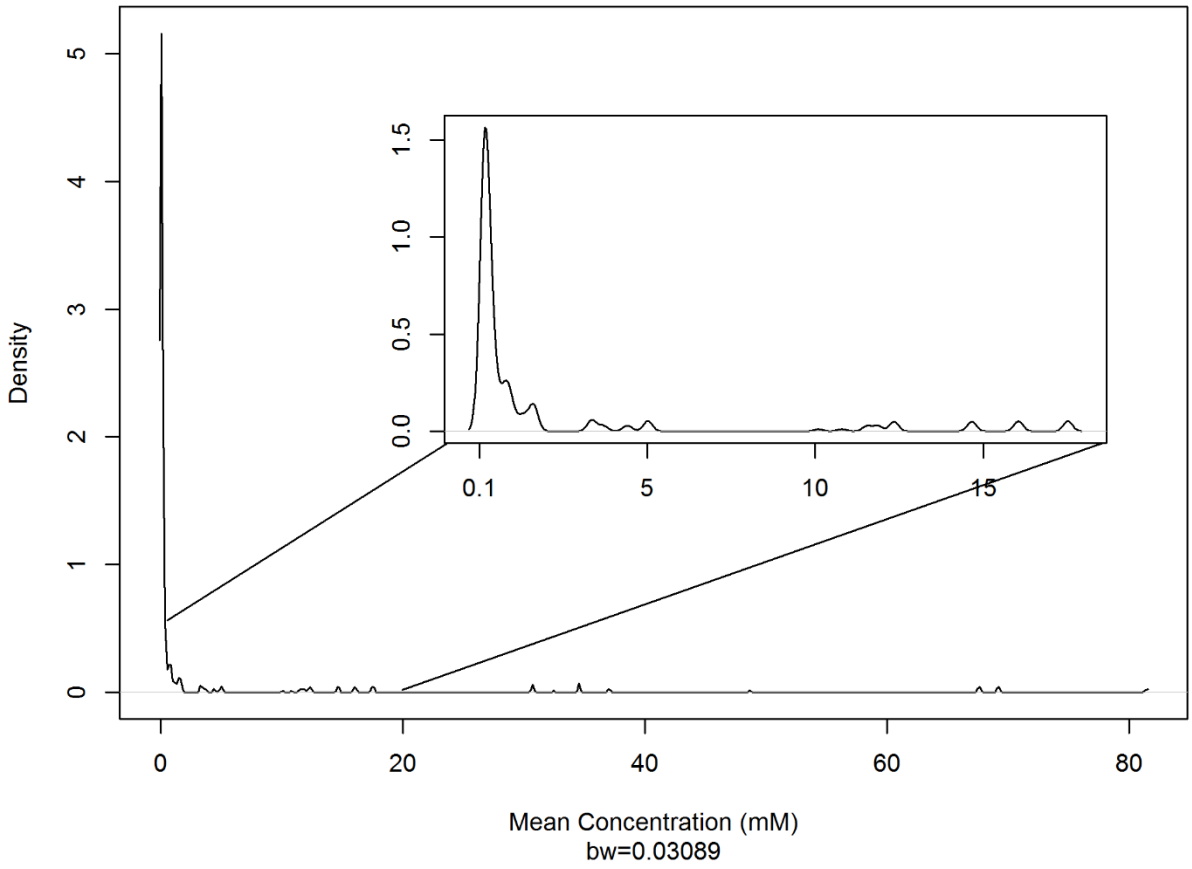

Figure A7. Kernel density plot of mean concentrations of untreated samples collected during steady state, bandwidth=0.03043. Inset is the kernel density plot of mean concentrations greater than or equal to 0.1 mM and less than or equal to 20 mM, bandwidth = 0.135. The clusters observed in these plots were used to define the bin in which the compounds would be grouped.

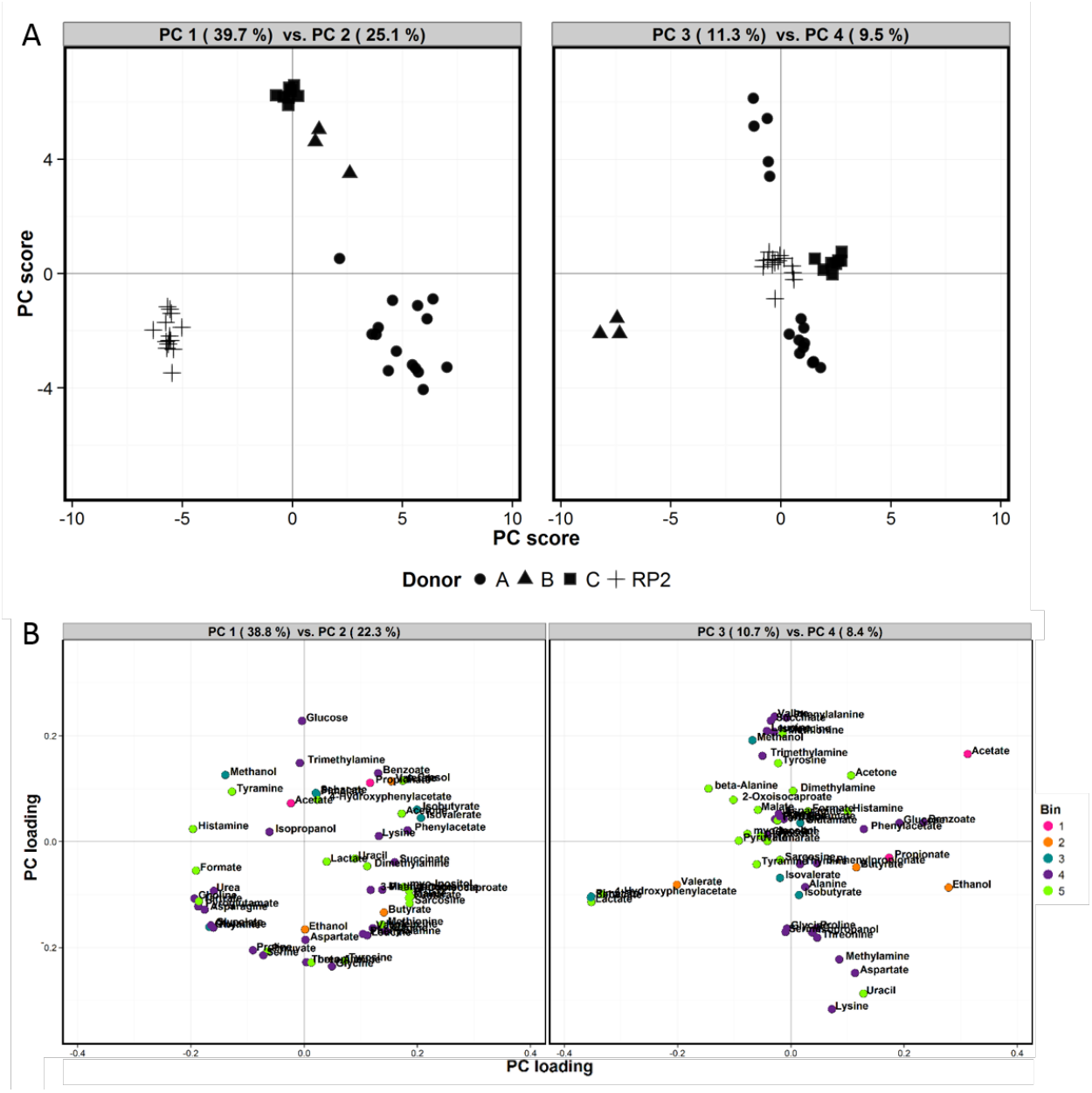

Figure A8. (A) PCA Score plot of metabolite profiles of donor A, B, C and MET-2 cultures. (B) PCA loading plot of metabolite profiles of donor A, B C and MET-2 cultures. Data is mean centered and scaled by unit variance. Bin 1: mean ≥ 20 mM; Bin 2: 10 < mean < 20 mM; Bin 3: 1 < mean < 10 mM; Bin 4: 0.1 < mean < 1 mM; Bin 5: 0 < mean < 0.

A.6 Analysis of Perturbations to Human Fecal Microbiota Propagated In Vitro

Table A6. List of microorganisms sensitive to clindamycin. Adapted from Spizek et al, 2004[216]

Organism	Minimum Inhibitory concentration (µg/mL)
Gram-positive	
Bacillus anthracis	0.25–5.0
Staphylococcus aureus	0.04–1.6
Sta. epidermidis	0.1–0.2
Streptococcus agalactiae	0.02–0.1
Str. pneumoniae	0.002–0.1
Str. pyogenes	0.01–0.2
Str. viridans	0.005–0.2
Gram-negative	
Escherichia coli	64
Haemophilus influenzae	0.5–16.0
Klebsiella pneumoniae	125
Neisseria gonorhoeae	0.5–4.0
N. meningitis	4
Proteus vulgaris	250
Pseudomonas aeruginosa	1000
Salmonella schottmuelleri	64

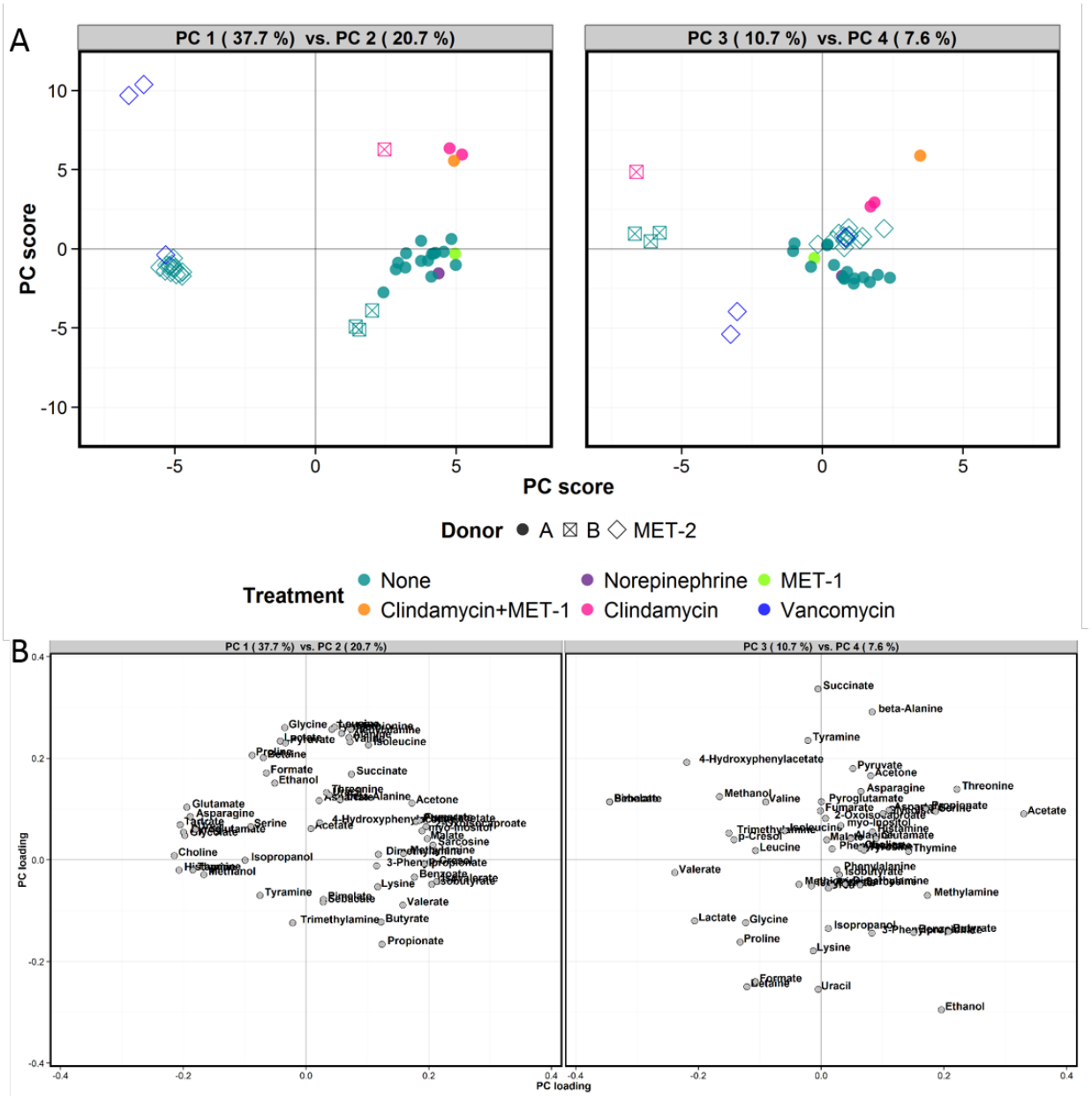

Figure A9. PCA model of metabolite profiles of Donor A, B, and MET-2, as well as profiles of each respective community culture after various treatments. (A) Score plot. (B) Loading plot.

Appendix B Calculations

Appendix B1. Relative Standard Deviation (RSD)

$$RSD = \frac{s}{\bar{x}} \times 100 \qquad ; \qquad s = \sqrt{\frac{\sum(x_i - \bar{x})^2}{n-1}}$$

Appendix B2. Equation for multiple linear regression.

The following equation was used to model each compound:

$$y_i = \beta_1 x_{i1} + \beta_2 x_2 + \beta_3 x_3 + \beta_3 x_1 x_2 + \beta_2 x_1 x_3 + intercept + \varepsilon$$

Where x_1 is ultracentrifuge speed, x_2 is experiment, x_3 is type of pre-processing step used, ε is the error and $i = 1 \ldots n$ samples.

Appendix B3. Bonferroni correction on statistical significance when performing multiple comparisons.

$$\alpha_{corrected} = \frac{\alpha}{n}$$

Where *n* is the number of times the statistical test is applied.

Appendix B4. Percent change in concentration per 10 000 rpm

$$percent\ change = \left(\frac{\beta_1 * 10000}{y_{10000}}\right) \times 100$$

Where the response at rpm=10000, y_{10000}, and coefficient, β_1, are from the MLR equation presented in Appendix B2.

Appendix B5. Analysis of Variance (ANOVA) about the mean. $MS_{residual}$ = Mean square within group, MS_{among} = Mean square between groups, n_j=number of observations within the j^{th} group, q=number of groups.

$$\overline{x_{.j}} = \frac{1}{n_j}\sum_{i=1}^{n_j} x_{ij} \qquad ; \qquad \bar{x}_{global} = \frac{1}{n}\sum_{j=1}^{q}\sum_{i=1}^{n_j} x_{ij} \qquad ; \qquad n = \sum_{j=1}^{q} n_j$$

$$SS_{residual} = \sum_{i}\left(x_{ij} - \overline{x}_{.j}\right)^2 \qquad ; \qquad MS_{residual} = \frac{SS_{residual}}{n_j - q}$$

$$SS_{among} = \sum\left(\overline{x}_{.j} - \overline{x}_{global}\right)^2 \qquad ; \qquad MS_{among} = \frac{SS_{among}}{q - 1}$$

$$F\ value = \frac{MS_{residual}}{MS_{among}}$$

Appendix C Programming

C.1 Standardization of Fecal Water Samples and Analysis

Data source.R

```
#////////////////////////////////////////////////////////////////////////////////
/
#Data source

require(RSQLite)
require(reshape2)

setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
source('function-summary preprocessing.R')

setwd('C:/Users/Sandi/Dropbox/Projects/Fecal water samples/R Scripts/Data source
    files')

#
    ------------------------------------------------------------------------------


raw <- summary.preprocess('C:/Users/Sandi/Dropbox/Projects/Fecal water samples/R
    Scripts/Data source files/concentrations_x2-fecal water.csv')

d <- raw[['data']]
d.melt <- melt(d)
d.melt$value <- d.melt$value/0.9 #adjusting for sample dilution with 10% DSS

raw2 <- summary.preprocess('C:/Users/Sandi/Dropbox/Projects/Fecal water samples/R
    Scripts/Data source files/concentrations_x1, x2, x3-fecal water.csv')

d2 <- raw2[['data']]
d2.melt <- melt(d2)
d2.melt$value <- d2.melt$value/0.9

raw3 <- summary.preprocess('C:/Users/Sandi/Dropbox/Projects/Fecal water samples/R
    Scripts/Data source files/concentrations-fecal samples_Eric_SY.csv')

d3 <- raw3[['data']]
d3.melt <- melt(d3)
d3.melt$value <- d3.melt$value/0.9

#Qualifier-------------------------------------------------------------------
qualifier <- data.frame(file=d$file)
qualifier$sampleID <- qualifier$file
qualifier$sampleID <- gsub('.*([A-Z])(?=_x).*', '\\1', qualifier$sampleID, perl=
    TRUE)

qualifier$scan <- qualifier$file
qualifier$scan <- gsub('.*(?<=[A-Z]_x)([0-9]{1,2}).*', '\\1', qualifier$scan, perl=
    TRUE)

qualifier$profile.no <- qualifier$file
qualifier$profile.no <- gsub('.*([0-9])(?=\\.cnx).*','\\1',
                             qualifier$profile.no, perl=TRUE)

qualifier$c.initial <- c(rep(8.2, 2), rep(5.6, 2), rep(9.1, 2), rep(20.6, 2),
                         rep(15.2, 2), rep(2.6, 2))
qualifier$v.initial <- c(rep(30,6), rep(60, 4), rep(30, 2))
qualifier$concentration <- round(qualifier$c.initial/qualifier$v.initial,4)

qualifier$rep <- c(rep(c(1,2),6))
qualifier$date <- 'June 5, 2014'

qualifier2 <- data.frame(file=d2$file)
qualifier2$sampleID <- qualifier2$file
qualifier2$sampleID <-  gsub('.*([A-Z])(?=_x).*', '\\1', qualifier2$sampleID, perl=
    TRUE)

```

```
qualifier2$scan <- qualifier2$file
qualifier2$scan <- gsub('.*(?<=[A-Z]_x)([0-9]{1,2}).*', '\\1', qualifier2$scan,
    perl=TRUE)

qualifier2$profile.no <- qualifier2$file
qualifier2$profile.no <- gsub('.*([0-9])(?=\\.cnx).*','\\1',
                              qualifier2$profile.no, perl=TRUE)

qualifier2$c.initial <- c(rep(c(rep(8.2, 2), rep(5.6, 2), rep(9.1, 2), rep(20.6, 2)
    ,
                          rep(15.2, 2), rep(2.6, 2)),3))
qualifier2$v.initial <- c(rep(c(rep(30,6), rep(60, 4), rep(30, 2)),3))
qualifier2$concentration <- round(qualifier2$c.initial/qualifier2$v.initial,4)
qualifier2$rep <- c(rep(c(rep(c(1,2),6)),3))
#Compound rank-----------------------------------------------------------
rank <- read.csv('Compound rank-fecal water.csv')
rank <- rank[,1:4]

# Create the connection (no file needed) ---------------------------------

# Choose driver
drv = dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con = dbConnect(drv, 'data.db')

# Creating database tables
data.table <- dbWriteTable(con, 'Data', d.melt, overwrite=TRUE)
data.table

data2.table <- dbWriteTable(con, 'Data2', d2.melt, overwrite=TRUE)
data2.table

data3.table <- dbWriteTable(con, 'Data3', d3.melt, overwrite=TRUE)
data3.table

qualifier.table <- dbWriteTable(con, 'Qualifier', qualifier, overwrite=TRUE)
qualifier.table

qualifier2.table <- dbWriteTable(con, 'Qualifier2', qualifier2, overwrite=TRUE)
qualifier2.table

rank.table <- dbWriteTable(con, 'Rank', rank, overwrite=TRUE)
rank.table

tables <- dbGetQuery(con, "SELECT * FROM sqlite_master WHERE type='table';")
print(tables)

dbDisconnect(con)
```

Fecal concentration analysis.R

```
#////////////////////////////////////////////////////////////////////////////
    //
#Analysis

#loading required packages
require(ggplot2)
require(reshape2)
require(RSQLite)

#sourcing required functions
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
source('function-data cleanup.R')
source('function-descriptive statistics.R')

setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation samples/Experiment 2/R
    Scripts/Functions')
```

```
source('correlation.R')

setwd('C:/Users/Sandi/Dropbox/Projects/Fecal water samples/R Scripts/Analysis')

source('../Functions/linear model.R')

#-----------------------------------------------------------------------------
#Input data
# Choose driver
drv <- dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

input <- dbGetQuery(con, "SELECT q.SampleID, q.concentration, q.rep,
                          d.variable, d.value
                          FROM Data d INNER JOIN Qualifier q
                          ON d.file = q.file
                          INNER JOIN Rank r
                          ON d.variable = r.Compound
                          WHERE r.Confidence = 1
                          ORDER BY q.sampleID;")


dbDisconnect(con)

#Data clean up-----------------------------------------------------------------
clean <- data_cleanup(input, convert.NA=TRUE)
clean$rep <- as.character(clean$rep)

#Descriptive statistics--------------------------------------------------------
stat <- descriptive.stat(clean, data.type='long', variable=c('variable','
    concentration'))

#Overview----------------------------------------------------------------------
p <- ggplot(stat, aes(x=concentration, y=value)) +
  geom_point(aes(colour=rep), alpha=0.8) +
  xlab('Fecal water concentration (g/mL)') +
  ylab('Compound concentration (mM)') +
  facet_wrap(~variable, scales='free', ncol=6) +
  theme_bw(14) +
  theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45))

ggsave('../Plots/Overview_x2.png', p, height=30, width=30, units='cm')

#Linear relationship-----------------------------------------------------------
lm.test <- ddply(stat, 'variable', lm_custom, group='concentration', value='avg')
lm.sig <- lm.test[lm.test$slope.p <=0.05,]

#correlation
corr.result <- ddply(stat, 'variable', corr_custom, x='concentration', y='avg')
corr.sig <- corr.result[corr.result$p.val <= 0.05,]

plot.stat <- cbind(lm.test, corr.result$estimate)
colnames(plot.stat)[10] <- 'corr.estimate'
plot.stat$linetype <- 'ns'
plot.stat$linetype[plot.stat$slope.p <= 0.05 &
                     plot.stat$corr.estimate > 0.60] <- 's'

p.lm <- ggplot(stat, aes(x=concentration, y=value)) +
  #geom_point(aes(colour=rep), alpha=0.6) +
  geom_point(aes(y=avg), colour='black', alpha=0.8) +
  geom_abline(data=plot.stat, aes(intercept=int, slope=slope,linetype=linetype)) +
  scale_linetype_manual(values=c(2,1)) +
  xlab('Fecal concentration (g/mL)') +
  ylab('Compound concentration (mM)') +
  facet_wrap(~variable, scales='free', ncol=6) +
  theme_bw(9) +
```

```
  theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45))

ggsave('../Plots/Relationship_x2_c.png', height=25, width=22, units='cm')

#Lowest concentrations----------------------------------------------------------
low <- stat[stat$concentration == 0.3433,]
low.cmpd <- unique(low[,c('variable','avg')])
low.cmpd <- low.cmpd[order(low.cmpd$avg),]
```

Normalization analysis.R

```
#////////////////////////////////////////////////////////////////////////////////
#normalizing for fecal water concentration

#loading required packages
require(ggplot2)
require(reshape2)
require(RSQLite)
require(gridExtra)

#sourcing required functions
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
source('function-data cleanup.R')
source('function-descriptive statistics.R')

setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation samples/Experiment 2/R Scripts/Functions')
source('correlation.R')

setwd('C:/Users/Sandi/Dropbox/Projects/Fecal water samples/R Scripts/Analysis')

source('../Functions/linear model.R')
source('../Functions/stat custom.R')
source('../Functions/normalize.R')
#-------------------------------------------------------------------------------
#Input data
# Choose driver
drv <- dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

input <- dbGetQuery(con, "SELECT q.SampleID, q.concentration, q.rep,
                          d.variable, d.value
                          FROM Data d INNER JOIN Qualifier q
                          ON d.file = q.file
                          INNER JOIN Rank r
                          ON d.variable = r.Compound
                          WHERE r.Confidence = 1
                          ORDER BY q.sampleID;")

input2 <- dbGetQuery(con, "SELECT * FROM Data3 d3 INNER JOIN Rank r
                           ON d3.variable = r.Compound
                           WHERE r.Confidence = 1;")

dbDisconnect(con)

#Data clean up------------------------------------------------------------------
clean <- data_cleanup(input, convert.NA=TRUE)
clean$rep <- as.character(clean$rep)

#export in excel
cast <- dcast(clean, sampleID+concentration+rep~variable, value.var='value')
#write.csv(file='C:/Users/Sandi/Dropbox/Projects/Fecal water samples/R Scripts/Results/fecal water data.csv', cast)

#Descriptive statistics---------------------------------------------------------
```

```
stat <- descriptive.stat(clean, data.type='long',
                         variable=c('variable','concentration'))

#Linear relationship between metabolite concentration with fecal water
    concentration----------
lm.test <- ddply(stat, 'variable', lm_custom, group='concentration', value='avg')
lm.sig <- lm.test[lm.test$slope.p <=0.05,]

#correlation
corr.result <- ddply(stat, 'variable', corr_custom, x='concentration', y='avg')
corr.sig <- corr.result[corr.result$p.val <= 0.05,]

plot.stat <- cbind(lm.test, corr.result$estimate)
colnames(plot.stat)[10] <- 'corr.estimate'
plot.stat$linetype <- 'ns'
plot.stat$linetype[plot.stat$slope.p <= 0.05 &
                     plot.stat$corr.estimate > 0.60] <- 's'

p.lm <- ggplot(stat, aes(x=concentration, y=value)) +
  geom_point(aes(colour=rep), alpha=0.6) +
  geom_point(aes(y=avg), colour='black', alpha=0.8) +
  geom_abline(data=plot.stat, aes(intercept=int, slope=slope,linetype=linetype)) +
  scale_linetype_manual(values=c(2,1)) +
  xlab('Fecal water concentration (g/mL)') +
  ylab('Compound concentration (mM)') +
  facet_wrap(~variable, scales='free', ncol=6) +
  theme_bw(14) +
  theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45))

#Normalizing metabolite concentration for fecal water concentration----------
#experssing compound concentration as per g of feces
clean$norm <- clean$value/clean$concentration/1000

#Do the standardized concentrations have slope significantly different than zero
lm.norm <- ddply(clean, 'variable',lm_custom, group='concentration',value='norm')
lm.norm$linetype <- 'ns'
lm.norm$linetype[lm.norm$slope.p <=0.05] <- 's'

#Visualizing normalized values----------------------------------------------
p <- ggplot(clean, aes(x=concentration, y=norm)) +
  geom_point(aes(colour=rep), alpha=0.8) +
  geom_abline(data=lm.norm, aes(intercept=int, slope=slope, linetype=linetype)) +
  scale_linetype_manual(values=c(2,1)) +
  xlab('Fecal water concentration (g/mL)') +
  ylab('Standardized Compound concetration (mmol/g feces)') +
  facet_wrap(~variable, scales='free', ncol=6) +
  theme_bw(14) +
  theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45))

#ggsave('../Plots/Normalized Overview.png', p, height=30, width=30, units='cm')


#looking at what the slopes of normalized values are
png('../Plots/Normalized slopes.png', units='cm', height=8, width=12, res=240)
plot(density(lm.norm$slope), main='', sub='N=39, Bandwidth=0.001',
     xlab='Slope (mM/g of feces)')
dev.off()


p.norm <- ggplot(norm.stat, aes(x=variable, y=norm)) +
  stat_boxplot() +
  geom_point() +
  xlab('Compound') +
  ylab('Normalized Compound Concentration (mmol/g feces)') +
  facet_wrap(~variable, scales='free',ncol=5) +
  theme_bw(14) +
  scale_x_discrete(labels=NULL) +
  theme(axis.ticks.x=element_blank())
```

```
122
123 ggsave('../Plots/Normalized concentration_b.png', p.norm,
124        height=35, width=30, units='cm')
125
126 #Normalizing metabolite concentration given mM vs g/mL relationship as a score
    ----------
127 clean$slope <- NA
128 for(i in lm.test$variable) {
129   s <- lm.test$slope[lm.test$variable==i]
130   clean$slope[clean$variable == i] <- s
131 }
132
133 clean$score <- clean$value/clean$slope/clean$concentration/1000
134
135 #Do the normalized scores have slope significantly different than zero
136 lm.score <- ddply(clean, 'variable',lm_custom, group='concentration',value='score')
137 lm.score$linetype <- 'ns'
138 lm.score$linetype[lm.score$slope.p <=0.05] <- 's'
139
140
141 p <- ggplot(clean, aes(x=concentration, y=score)) +
142   geom_point(aes(colour=rep), alpha=0.8) +
143   geom_abline(data=lm.score, aes(intercept=int, slope=slope, linetype=linetype)) +
144   scale_linetype_manual(values=c(2,1)) +
145   xlab('Fecal water concentration (g/mL)') +
146   ylab('Normalized Score of Compound Concetration') +
147   facet_wrap(~variable, scales='free', ncol=6) +
148   theme_bw(14) +
149   theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45))
150
151 #ggsave('../Plots/Normalized Score Overview.png', p, height=30, width=30, units='cm
    ')
152
153 #Comparing Normalized profile to previous fecal sample profiles----------
154
155 clean2 <- data_cleanup(input2, convert.NA=TRUE)
156 clean2$rep <- as.character(clean$rep)
157 stat2 <- descriptive.stat(clean2, data.type='long', variable='variable')
158
159 stat2$scaled <- stat2$value/100
160 stat2$scaled[stat2$variable == 'Ethanol'] <- stat2$scaled[stat2$variable == '
    Ethanol']/50
161 stat2$expt <- 'Preliminary experiment'
162 norm.stat$expt <- 'Smallest Poop experiment'
163
164 pf <- ggplot(stat2, aes(x=variable, y=scaled, colour=expt), alpha=0.8) +
165   stat_boxplot(data=norm.stat, aes(y=norm)) +
166   geom_point(data=norm.stat, aes(y=norm)) +
167   stat_boxplot() +
168   geom_point() +
169   xlab('Compound') +
170   ylab('Compound Concentration') +
171   theme_bw(14) +
172   theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45),
173         legend.position='none')
174 #ggsave('../Plots/Old fecal samples.png', pf, height=15, width=35, units='cm')
175
176 #Normalizing by highest concentration compound (Acetate)----------
177 cmpd.norm <- ddply(stat, 'concentration', norm_custom,
178                 norm='Acetate', norm.val='avg',norm.name='norm.acetate')
179 cmpd.norm <- ddply(cmpd.norm, 'concentration', norm_custom,
180                 norm='Propionate', norm.val='avg',norm.name='norm.propionate')
181 cmpd.norm <- ddply(cmpd.norm, 'concentration', norm_custom,
182                 norm='Butyrate', norm.val='avg',norm.name='norm.butyrate')
183
184 norm <- melt(cmpd.norm, id.vars=c('sampleID','concentration','rep','variable'),
185              measure.vars=c('value','avg','norm.acetate','norm.propionate',
186                             'norm.butyrate'),
```

```
                variable.name='norm.type')

norm <- norm[!norm$norm.type %in% c('value','avg'),]

cmpd.stat <- ddply(norm, 'norm.type', descriptive.stat, data.type='long',
                   variable=c('variable','concentration'))

#Compare feces norm and acetate normalization----------------------------------
#Getting descriptive stats of the normalized values
stat_func <- function(d, value){
  stdev <- sd(d[,value])
  avg <- mean(d[,value])
  ymax <- max(d[,value])
  ymin <- min(d[,value])
  se <- stdev/sqrt(nrow(d))
  out <- data.frame(avg=avg, stdev=stdev, se=se, ymax=ymax, ymin=ymin)
  return(out)
}

fece.stat <- ddply(clean, 'variable', stat_func, value='norm')
acet.stat <- ddply(cmpd.norm, 'variable', stat_func, value='norm.acetate')

diff.stat <- data.frame(variable=fece.stat$variable, fec.sd=fece.stat$stdev,
                        fec.se=fece.stat$se, ace.sd=acet.stat$stdev,
                        ace.se=acet.stat$se)
diff.stat$diff.se <- diff.stat$fec.se-diff.stat$ace.se
diff.stat$diff.sd <- diff.stat$fec.sd-diff.stat$ace.sd

fece.stat$norm.type <-'fec'
acet.stat$norm.type <- 'norm.acetate'

norm.stat <- rbind(fece.stat, acet.stat)

#Comparing Normalized profile to non-norm values-------------------------------
p.data <- clean
p.data$scal.val <- p.data$value/100
p.data <- melt(p.data, measure.vars=c('value','norm','scal.val'))
colnames(p.data)[5] <- 'data.type'
p.data <- p.data[p.data$data.type != 'value',]
#p.data <- merge(p.data, norm.stat[norm.stat$norm.type=='fec',], by='variable')

p.feces <- ggplot(p.data, aes(x=variable, y=value, colour=data.type)) +
  stat_boxplot() +
  #geom_point(position=position_dodge(width=1)) +
  ylab('Compound Concentration') +
  theme_bw(8) +
  theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45),
        axis.title.x=element_blank(),
        legend.position=c(0.5,-0.35),
        legend.direction='horizontal',
        legend.margin=unit(0,'cm'),
        panel.margin=unit(c(0.3,0.1,0,0.1),'cm'))

plot.data <-cmpd.stat[cmpd.stat$norm.type=='norm.acetate',]
p.acetate <- ggplot(plot.data, aes(x=variable, y=value)) +
  stat_boxplot(aes(ymax=ymax, ymin=ymin)) +
  #geom_point() +
  theme_bw(8) +
  xlab('Compound') +
  ylab('Compound concentration (% Acetate)') +
  theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45),
        panel.margin=unit(c(0,0.1,0.2,0.1),'cm'))

g <- arrangeGrob(p.feces, p.acetate, ncol=1, nrow=2, height=c(0.5,0.5))

png(file='../Plots/norm compare.png',width=18,height=18, units='cm',res=300)
print(g)
grid.text('A', x=0.0345,y=0.98)
```

```
grid.text('B', x=0.0345,y=0.51)
dev.off()

#Inner join two normalizations into one df for plotting-------------------------
p.data <- data.frame(variable=clean$variable,
                     concentration=clean$concentration,
                     rep=clean$rep,
                     fec=clean$norm)

p.data <- merge(p.data, cmpd.norm[,c('variable','concentration','rep',
                                     'norm.acetate')],
                by=c('variable','concentration','rep'))

p.data <- melt(p.data, id.vars=c('variable','concentration','rep'),
               measure.vars=c('fec','norm.acetate'))

colnames(p.data)[4:5] <- c('norm.type','norm.val')

p.data <- merge(p.data, norm.stat, by=c('variable','norm.type'))

p.fec <- ggplot(p.data[p.data$norm.type=='fec',], aes(x=variable, y=norm.val)) +
  geom_pointrange(aes(y=avg, ymin=ymin, ymax=ymax), size=0.3)+
  #stat_boxplot() +
  theme_bw(10) +
  #xlab('Compound') +
  ylab('Nomalized concentration (mM/g feces)') +
  theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45),
        axis.title.x=element_blank(),
        plot.margin=unit(c(0.8,0.5,0,0.2),"cm"))

#ggsave('../Plots/norm range-feces.png', p.fec, height=11.5, width=18, units='cm')

p.ace <- ggplot(p.data[p.data$norm.type=='norm.acetate',], aes(x=variable, y=norm.
    val)) +
  geom_pointrange(aes(y=avg, ymin=ymin, ymax=ymax), size=0.3)+
  #stat_boxplot() +
  theme_bw(10) +
  xlab('Compound') +
  ylab('Nomalized concentration (% acetate)') +
  theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45),
        plot.margin=unit(c(0,0.5,0.2,0.2),"cm"))

#ggsave('../Plots/norm range-acetate.png', p.ace, height=11.5, width=18, units='cm
    ')

g <- arrangeGrob(p.fec, p.ace, ncol=1, nrow=2, height=c(0.5,0.5))

png(file='../Plots/norm range.png',width=15,height=18, units='cm',res=300)
print(g)
grid.text('A', x=0.0345,y=0.98)
grid.text('B', x=0.0345,y=0.51)
dev.off()
#Compare difference compound normalizations-------------------------------------

p.norm <- ggplot(cmpd.stat, aes(x=variable, y=value, colour=norm.type)) +
  stat_boxplot(aes(ymax=ymax, ymin=ymin)) +
  geom_point(position=position_dodge(width=0.8)) +
  theme_bw(14) +
  theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45))

ggsave('../Plots/Normalize by diff compounds.png', p.norm,
       height=15, width=35, units='cm')

#plotting only acetate normalized and poop nomalized
plot.data <-cmpd.stat[cmpd.stat$norm.type=='norm.acetate',]
plot.data2 <- norm.stat
plot.data2$norm2 <- plot.data2$norm*25

```

```
p.acetate <- ggplot(plot.data, aes(x=variable, y=value)) +
  #stat_boxplot(data=plot.data2, aes(y=norm2), colour='grey60') +
  #geom_point(data=plot.data2, aes(y=norm2), colour='grey60') +
  stat_boxplot(aes(ymax=ymax, ymin=ymin)) +
  geom_point() +
  theme_bw(14) +
  xlab('Compound') +
  ylab('Compound concentration (% Acetate)') +
  theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45))

ggsave('../Plots/Normalize by acetate.png', p.acetate,
       height=15, width=35, units='cm')

pa2 <- ggplot(plot.data, aes(x=variable, y=value)) +
  stat_boxplot(aes(ymax=ymax, ymin=ymin)) +
  geom_point() +
  facet_wrap(~variable, scales='free', ncol=5) +
  theme_bw(14)+
  xlab('Compound') +
  ylab('Normalized Compound Concentration (% Acetate)') +
  scale_x_discrete(labels=NULL) +
  theme(axis.ticks.x=element_blank())

ggsave('../Plots/Normalize by acetate-concentrations2.png', pa2,
       height=35, width=30, units='cm')
```

C.2 Gut-mimicking CSTR Effluent Sample Preparation and Analysisprep protocol

Data source.R

```
#////////////////////////////////////////////////////////////////////////////
#Data source for UC Expt 3


#sourcing required functions
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
source('function-summary preprocessing.R')

#loading required packages
require(reshape2)
require(RSQLite)

#Setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 3/R
    Scripts/Data source files')

#preprocessing raw data-------------------------------------------------------
#Experiment 1
raw1 <- summary.preprocess('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation
    Samples/Experiment 1/Results/R Scripts/Data source files/concentrations-UC Expt
     1.csv')
data1 <- raw1[['data']]
data.melt1 <- melt(data1)
data.melt1$value <- data.melt1$value/0.9

mw <- raw1[['mol.wt']]
mw1 <- as.data.frame(t(mw))
mw1$variable <- rownames(mw1)
mw1$Weight <- as.numeric(as.character(mw1$Weight))


#Experiment 2
```

```
raw2 <- summary.preprocess('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation
    samples/Experiment 2/R Scripts/Data source files/concentrations_x2-UC Expt 2.
    csv')
data2 <- raw2[['data']]
data.melt2 <- melt(data2)
data.melt2$value <- data.melt2$value/0.9

mw <- raw2[['mol.wt']]
mw2 <- as.data.frame(t(mw))
mw2$variable <- rownames(mw2)
mw2$Weight <- as.numeric(as.character(mw2$Weight))

#Experiment 3
raw3 <- summary.preprocess('concentrations-UC Expt 3.csv')
data3 <- raw3[['data']]
data.melt3 <- melt(data3)
data.melt3$value <- data.melt3$value/0.9

mw <- raw1[['mol.wt']]
mw3 <- as.data.frame(t(mw))
mw3$variable <- rownames(mw3)
mw3$Weight <- as.numeric(as.character(mw3$Weight))

#Preliminary experiment
raw.prelim <- summary.preprocess('C:/Users/Sandi/Dropbox/Projects/
    Ultracentrifugation samples/Preliminary Experiment/R scripts/Data source files/
    D5-Ultracentrifugation concentration summary_2.csv')
data.prelim <- raw.prelim[['data']]
prelim.melt <- melt(data.prelim)
prelim.melt$value <- prelim.melt$value/0.9

mw <- raw1[['mol.wt']]
mw.prelim <- as.data.frame(t(mw))
mw.prelim$variable <- rownames(mw.prelim)
mw.prelim$Weight <- as.numeric(as.character(mw.prelim$Weight))

data <- rbind(data.melt1, data.melt2, data.melt3, prelim.melt)

mw <- unique(rbind(mw1, mw2, mw3, mw.prelim))
#Qualifiers-------------------------------------------------------------------
#file == file name from batch export
#scan == scan number within Expt
#sample == sample ID within UC Expt
#sampleID == sample ID within UC project -- unique across expts
#type == grouping triplicate samples within UC Expt -- not unique across expts
#grp == grouping triplicate sampels across project (UC Expt 1 and 2 combined)
#          -- unique across expts
#rep == replicate number
#class == processing procedure used
#speed == speed at which samples was ultracentrifuged
#process == presence of pre spin or no
#filter == filter size
#expt == experiment number

#Experiment 1-----------------------------------------------------------------
qualifier1 <- data.frame(file=data1$file)

qualifier1$scan <- qualifier1$file
qualifier1$scan <- gsub('.*(?<=x)([0-9]{1,2}).*', '\\1',
                        qualifier1$scan, perl=TRUE)

scan <- read.csv('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/
    Experiment 1/Results/R Scripts/Data source files/scan order.csv', header=FALSE)
scan <- scan[2:nrow(scan),]
col.name <- as.character(as.matrix(scan[1,]))
colnames(scan) <- col.name
scan <- scan[2:nrow(scan),]
scan[,'Scan Order'] <- as.numeric(scan[,'Scan Order'])
```

```
scan$sampleID <- as.character(scan$sample)

type <- c()
for(i in 1:9) {
  temp <- rep(LETTERS[i], 3)
  type <- c(type, temp)
}

grp <- c()
for(i in 1:18) {
  temp <- rep(LETTERS[i], 3)
  grp <- c(grp, temp)
}

scan$type <- type
scan$grp <- grp[1:nrow(scan)]
scan$rep <- c(rep(1:3, 9))
scan$class <- c(rep('I',15), rep('II',6), rep('III',6))
scan$speed <- c(rep(10000,3), rep(20000,3), rep(30000,3), rep(40000,3), rep(50000,3),
                rep(10000,3), rep(50000,3),rep(0,6))
scan$process <- c(rep('pre-spin',15), rep('no pre-spin',6), rep('pre-spin',6))
scan$filter <- c(rep('0.22',21), rep('serial',3),rep('0.22',3))

scan <- scan[order(scan[,'Scan Order']),]

qualifier1 <- cbind(qualifier1, scan)
qualifier1 <- qualifier1[,!colnames(qualifier1) %in% c('sample_name',
                                                       'Scan Order')]

qualifier1$expt <- 'expt1'
qualifier1$sampleID <- factor(qualifier1$sampleID,
                              levels=c(LETTERS[1:26],
                                       paste(LETTERS[1],LETTERS[1],sep='')))
qualifier1 <- qualifier1[order(qualifier1$sampleID),]

#Experiment 2-----------------------------------------------------------------
qualifier2 <- data.frame(file=data2$file)
qualifier2$scan <- qualifier2$file
qualifier2$scan <- gsub('.*([0-9]{2})(?=_x2\\.cnx).*', '\\1', qualifier2$scan, perl
    =TRUE)
qualifier2$scan <- gsub('.*([0-9]{1})(?=_x2\\.cnx).*', '\\1', qualifier2$scan, perl
    =TRUE)
qualifier2$sample <- qualifier2$file
qualifier2$sample <- gsub('([A-Z]{1,2})(?=_x[0-9]{1,2}_).*','\\1',
                          qualifier2$sample, perl=TRUE)

qualifier2$sample <- factor(qualifier2$sample,
                            levels=c(LETTERS[1:26],
                                     paste(LETTERS[1],LETTERS[1],sep='')))
qualifier2 <- qualifier2[order(qualifier2$sample),]


sampleID <- c(paste(LETTERS[1],LETTERS[2:26], sep=''),
              paste(LETTERS[2],LETTERS[1:26], sep=''),
              paste(LETTERS[3],LETTERS[1:26], sep=''))

qualifier2$sampleID <- sampleID[1:nrow(qualifier2)]

qualifier2$type <- type
qualifier2$grp <- grp[nrow(scan)+1:nrow(qualifier2)]
qualifier2$rep <- c(rep(1:3, 9))
qualifier2$class <- c(rep('I',15), rep('II',6), rep('III',6))
qualifier2$speed <- c(rep(10000,3), rep(20000,3), rep(30000,3), rep(40000,3), rep(50000,3),
                      rep(10000,3), rep(50000,3),rep(0,6))
qualifier2$process <-  c(rep('pre-spin',15), rep('no pre-spin',6), rep('pre-spin',6))
```

```
qualifier2$filter <- c(rep('0.22',21), rep('serial',3),rep('0.22',3))
qualifier2$expt <- 'expt2'

#Experiment 3-----------------------------------------------------------------
qualifier3 <- data.frame(file=data3$file, scan=data3$file)
qualifier3$scan <- gsub('.*(?<=_x)([0-9]{1,2}).*', '\\1', qualifier3$scan,
                        perl=TRUE)

scan.order <- read.csv('scan order.csv')
scan.order <- scan.order[,2:3]

qualifier3$sample <- scan.order[,2]
qualifier3 <- qualifier3[order(qualifier3$sample),]

qualifier3$sampleID <- sampleID[(nrow(qualifier2)+1):
                                (nrow(qualifier2)+nrow(qualifier3))]

qualifier3$type <- c(rep('A',8), unlist(lapply(LETTERS[2:6], rep, 3)))
qualifier3$grp <- c(rep('S',8), unlist(lapply(LETTERS[20:24], rep, 3)))
qualifier3$rep <- c(1:8, rep(1:3,5))
qualifier3$class <- c(rep('III',8), rep('I',15))

sample.info <- lapply(seq(50000, 10000, -10000), rep, times=3)
qualifier3$speed <- c(rep(0,8), unlist(sample.info))

qualifier3$process <- 'no pre-spin'
qualifier3$filter <- c(rep('serial',8), rep(0.22,15))
qualifier3$expt <- 'expt3'

#Preliminary experiment-------------------------------------------------------
qualifier.prelim <- data.frame(file=data.prelim$file)
qualifier.prelim$scan <- c(1,2)
qualifier.prelim$sample <- LETTERS[1:nrow(qualifier.prelim)]
qualifier.prelim$sampleID <- sampleID[(nrow(qualifier2)+nrow(qualifier3)+1):
                                      (nrow(qualifier2)+nrow(qualifier3)+
                                        nrow(qualifier.prelim))]
qualifier.prelim$type <- LETTERS[1:length(unique(data.prelim$file))]
qualifier.prelim$grp <- LETTERS[(length(unique(qualifier1$grp))+
                                   length(unique(qualifier2$grp))+1):
                                  (length(unique(qualifier1$grp))+
                                     length(unique(qualifier2$grp))+
                                     length(unique(qualifier.prelim$file)))]
qualifier.prelim$rep <- rep(1,2)
qualifier.prelim$class <- c('II','III')
qualifier.prelim$speed <- c(50000, 0)
qualifier.prelim$process <- c('no pre-spin','pre-spin')
qualifier.prelim$filter <- c(0.22, 'serial')
qualifier.prelim$expt <- 'preliminary'

#----------------------------------------------------------------------------
qualifier <- rbind(qualifier1, qualifier2, qualifier3, qualifier.prelim)
qualifier$sampleID <- as.character(qualifier$sampleID)

#Rank-------------------------------------------------------------------------
#Experiment 1
rank1 <- read.csv('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/
    Experiment 1/Results/R Scripts/Data source files/compound ranking.csv',
    encoding='UTF-8')
rank1 <- rank1[,1:4]
rank1$expt <- 'expt1'

#Experiment 2
rank2 <- read.csv('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation samples/
    Experiment 2/R Scripts/Data source files/Compound rank-UC Expt 2.csv', encoding
    ='UTF-8')
rank2 <- rank2[,1:4]
rank2$expt <- 'expt2'

```

```
#Experiment 3
rank3 <- read.csv('Compound rank-UC Expt 3.csv')
rank3 <- rank3[,1:4]
rank3$expt <- 'expt3'

#Preliminary experiment
rank.prelim <- read.csv('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation
    samples/Preliminary Experiment/R scripts/Data source files/Compound annotation.
    csv', encoding='UTF-8')
rank.prelim <- rank.prelim[,c(1:3,5)]
rank.prelim$expt <- 'preliminary'

rank <- rbind(rank1, rank2, rank3, rank.prelim)

#Compound info-----------------------------------------------------------------
info <- read.csv('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Data
    source files/Database-Compound classes.csv')

# Create the connection (no file needed) ---------------------------------------

# Choose driver
drv = dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con = dbConnect(drv, 'data.db')

# Creating database tables
uc.table <- dbWriteTable(con, 'UC3_data', data, overwrite=TRUE)
uc.table

qualifier.table <- dbWriteTable(con, 'Qualifier', qualifier, overwrite=TRUE)
qualifier.table

rank.table <- dbWriteTable(con, 'Rank', rank, overwrite=TRUE)
rank.table

info.table <- dbWriteTable(con, 'Cmpd_info', info, overwrite=TRUE)
info.table

mw.table <- dbWriteTable(con, 'Mol_wt', mw, overwrite=TRUE)
mw.table

tables <- dbGetQuery(con, "SELECT * FROM sqlite_master WHERE type='table';")
print(tables)

dbDisconnect(con)
```

Profiling precision analysis.R

```
#////////////////////////////////////////////////////////////////////////////
#Ultracentrifuge Experiment 3 - Analysis of profiling variability

#Pre-amble---------------------------------------------------------------------
#Setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Data source
    files/')

#Sourcing required functions
source('../Functions/function-data cleanup.R')
source('../Functions/function-descriptive statistics.R')

setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 2/R
    Scripts/Functions')
source('./ANOVA-mean.R')
source('./kruskal wallis.R')
source('./linear model.R')
source('./tukey.R')
source('./correlation.R')
```

```
18
19 #loading required packages
20 require(RSQLite)
21 require(reshape2)
22 require(plyr)
23 require(ggplot2)
24
25 #setting working directory
26 setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 3/R
       Scripts/Analysis')
27 #-----------------------------------------------------------------------------
28 #Input data
29 # Choose driver
30 drv <- dbDriver('SQLite')
31
32 # Create connection (for SQLite, this is a file)
33 con <- dbConnect(drv, '../Data source files/data.db')
34
35 expt3 <-  dbGetQuery(con, "SELECT q.sampleID, q.type, q.grp, q.rep, q.class,
36                       q.speed, q.process, q.filter, q.expt, c.Type, m.Weight,
37                       d.variable, d.value
38                       FROM UC3_Data d INNER JOIN Qualifier q
39                       ON d.file = q.file
40                       INNER JOIN Rank r
41                       ON d.variable = r.Compound AND q.expt = r.expt
42                       INNER JOIN Cmpd_info c
43                       ON d.variable = c.Compound
44                       INNER JOIN Mol_wt m
45                       ON d.variable = m.variable
46                       WHERE r.Confidence = 1 AND q.expt = 'expt3'
47                       AND q.type = 'A'
48                       AND NOT (d.variable = 'Formate' AND d.value > 0.3)
49                       ORDER BY q.type;")
50
51 dbDisconnect(con)
52
53 #Data clean up-----------------------------------------------------------------
54 clean <- data_cleanup(expt3, convert.NA=TRUE)
55
56 #Descriptive statistics--------------------------------------------------------
57 stat <- descriptive.stat(clean, data.type='long', variable=c('type','variable'))
58
59 #Relative standard deviation---------------------------------------------------
60
61 stat$rsd <- stat$stdev/stat$avg*100
62
63 plot(density(stat$rsd, bw=1))
64
65 png(filename='../Plots/precision-RSD.png',
66     width=10, height=7, units='cm', res=240)
67 plot(density(stat$rsd, bw=1), main='')
68 dev.off()
69
70 high.rsd <- unique(stat[stat$rsd > 15,c('variable','rsd')])
71 write.csv(file='../Results/prescision-rsd.csv',high.rsd)
```

Factor effects-mlr analysis.R

```
1 ////////////////////////////////////////////////////////////////////////////
     //
2 #Comparison of profiles from Expt 3, amd Expt 2
3 #looking at effects of all explanatory variables (process, speed, expt)
4
5 #pre-amble-------------------------------------------------------------------
6 #Setting working directory
7 setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
8
9 #Sourcing required functions
```

```
source('function-data cleanup.R')
source('function-descriptive statistics.R')

setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 2/R
    Scripts/Functions')
source('./ANOVA-mean.R')
source('./kruskal wallis.R')
source('./linear model.R')
source('./tukey.R')
source('./correlation.R')

#loading required packages
require(RSQLite)
require(reshape2)
require(plyr)
require(ggplot2)

#setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 3/R
    Scripts/Analysis')
source('../Functions/multiple linear regression.R')
source('../Functions/mlr prediction.R')
#-------------------------------------------------------------------------------


#Input data
# Choose driver
drv <- dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

input <- dbGetQuery(con, "SELECT q.SampleID, q.type, q.grp, q.rep, q.class,
                      q.speed, q.process, q.filter, q.expt, c.Type, m.Weight,
                      d.variable, d.value
                      FROM UC3_Data d INNER JOIN Qualifier q
                      ON d.file = q.file
                      INNER JOIN Rank r
                      ON d.variable = r.Compound AND q.expt = r.expt
                      INNER JOIN Cmpd_info c
                      ON d.variable = c.Compound
                      INNER JOIN Mol_wt m
                      ON d.variable = m.variable
                      WHERE
                      (q.expt = 'expt2' AND r.Confidence = 1) OR
                      (q.expt = 'expt3' AND r.Confidence = 1)
                      AND NOT (d.variable = 'Formate' AND d.value > 0.3)
                      ORDER BY q.expt;")
dbDisconnect(con)

#Data clean up-----------------------------------------------------------------
clean <- data_cleanup(input, convert.NA=TRUE)

#Descriptive statistics--------------------------------------------------------
stat <- ddply(clean, 'expt', descriptive.stat, data.type='long',
              variable=c('type','variable'))

#Removing compounds found only in one of Expt 2 and Expt 3---------------------
expt2.var <- unique(stat$variable[stat$expt=='expt2'])
expt3.var <- unique(stat$variable[stat$expt=='expt3'])

remove <- expt2.var[!expt2.var %in% expt3.var]
sub <- stat[stat$variable != remove,]
#General to simplest mlr model-------------------------------------------------
check_model <- function(d, f1, f2) {
  m1 <- lm(formula=f1, data=d)
  m2 <- lm(formula=f2, data=d)
  compare <- anova(m1, m2)
```

```
  out <- compare[2,6]
  return(out)
}

bonferroni <- 0.05/length(unique(sub$variable))
mlr1 <-ddply(sub, 'variable', mlr_custom,
             f='value~speed+expt+process+speed:process+speed:expt')
mlr1$label <- ''
mlr1$label[mlr1[,'Pr(>|t|)'] <bonferroni] <- '*'
mlr1$model <- -0.3/2

mlr2 <- ddply(sub, 'variable', mlr_custom,
              f='value~speed+expt+process+speed:expt') #remove speed:process
mlr2$label <- ''
mlr2$label[mlr2[,'Pr(>|t|)'] <bonferroni] <- '*'
mlr2$model <- -0.1/2

#compare mlr1 and mlr2
m1m2 <-ddply(sub, 'variable', check_model,
             f1='value~speed+expt+process+speed:process+speed:expt',
             f2='value~speed+expt+process+speed:expt')

mlr3 <- ddply(sub, 'variable', mlr_custom,
              f='value~speed+expt+speed:expt') #remove process
mlr3$label <- ''
mlr3$label[mlr3[,'Pr(>|t|)'] <bonferroni] <- '*'
mlr3$model <- 0.1/2

m2m3 <- ddply(sub, 'variable', check_model,
              f1='value~speed+expt+process+speed:expt',
              f2='value~speed+expt+speed:expt')

mlr4 <- ddply(sub, 'variable', mlr_custom,
              f='value~speed+expt') #remove speed:expt
mlr4$label <- ''
mlr4$label[mlr4[,'Pr(>|t|)'] <bonferroni] <- '*'
mlr4$model <- 0.3/2

m3m4 <- ddply(sub, 'variable', check_model,
              f1='value~speed+expt+speed:expt',
              f2='value~speed+expt')

#visualizing significance of terms----------------------------------------------
plot.data <- rbind(mlr1, mlr2, mlr3, mlr4)
plot.data <- plot.data[plot.data$rname != '(Intercept)',]
plot.data$rname <- factor(plot.data$rname, levels=c('speed','exptexpt3',
                                                   'processpre-spin','speed:exptexpt3',
                                                   'speed:processpre-spin'))
var.level <- rev(unique(plot.data$variable))
plot.data$variable <- factor(plot.data$variable, levels=var.level)

rname.code <- data.frame(rname=unique(plot.data$rname),
                         rname.code=c(1,2,3,5,4))

plot.data <- merge(plot.data, rname.code, by='rname')
plot.data$x <- plot.data$model + plot.data$rname.code

var.code <- data.frame(variable=var.level)
var.code$var.code <- rev(as.numeric(var.code$variable))
plot.data <- merge(plot.data, var.code, by='variable')
plot.data$y <- plot.data$var.code
plot.data$var.code[plot.data$label != '*'] <- NA

plot.data$model <- factor(plot.data$model)

p <- ggplot(plot.data, aes(x=x, y=var.code)) +
  geom_point(aes(colour=model), shape=42, size=5) +
  scale_x_discrete(breaks=1:5, labels=c('Speed','Experiment','Pre-Process',
```

```
                                'Speed-Experiment\nInteraction',
                                'Speed-Pre-process\nInteraction')) +
  coord_cartesian(xlim = c(0.5, 5.5)) +
  scale_colour_discrete(name='MLR Model',labels=c(paste('model',1:4,sep=' '))) +
  scale_y_continuous(breaks=var.code$var.code,labels=var.code$variable) +
  xlab('Effect') +
  ylab('Compound') +
  theme_bw(11) +
  theme(axis.ticks.x=element_blank(),
        panel.grid.minor.x=element_blank(), panel.grid.major.x=element_blank(),
        panel.grid.minor.y=element_blank())

ggsave('../Plots/Factor effects-mlr analysis/simplifying mlr-model compare2.png', p,
       height=20, width=22, units='cm')

#Prediction from MLR-------------------------------------------------------------
mlr4.predict <- ddply(sub, 'variable', mlr_predict, f='value~speed+expt')

#absolute changes----------------------------------------------------------------
values <- unique(mlr4.predict[,c('speed','variable','avg','prediction')])

ran <- ddply(values, 'variable', mutate, r.avg1=range(avg)[1], r.avg2=range(avg)[2],
             r.pred1=range(prediction)[1], r.pred2=range(prediction)[2])
ran <- unique(ran)
ran$diff.avg <- ran$r.avg2-ran$r.avg1
ran$diff.pred <- ran$r.pred2-ran$r.pred1

diff <- unique(ran[ran$variable %in% c('Phenylalanine','Tyrosine','p-Cresol'),
                   c('variable','diff.avg','diff.pred')])

abs.slope <- mlr4[mlr4$variable %in% c('Phenylalanine','Tyrosine','p-Cresol'),]
#Perecent decrease per 10 000 rpm-----------------------------------------------
values <- unique(mlr4.predict[mlr4.predict$class=='I',
                              c('expt','speed','variable','avg','prediction')])
pred_custom <- function(d, formula){
  m <- lm(d, formula=formula)
  s <- summary(m)
  out <- d
  out$slope <- s$coefficients[2,1]
  out$int <- s$coefficients[1,1]
  return(out)
}
lm.pred <- ddply(values, c('variable','expt'), pred_custom,
                 formula="prediction~speed")

per.diff_custom <- function(d) {
  d$per.diff <- NA
  for(i in 2:nrow(d)) {
    d$per.diff[i] <- d$slope[i]*10000/d$prediction[d$speed==10000]*100
  }
  return(d)
}

lm.pred <- ddply(lm.pred, c('variable','expt'), per.diff_custom)

export <- unique(lm.pred[!is.na(lm.pred$per.diff),c('expt','variable','per.diff')])
#write.csv(file='../Results/Percent change per 10 000 rpm2.csv', export)

per.stat <- export[export$variable != 'p-Cresol',]
avg.per1 <- mean(per.stat$per.diff[per.stat$expt=='expt2'])
avg.per2 <- mean(per.stat$per.diff[per.stat$expt=='expt3'])
sd.per1 <- sd(per.stat$per.diff[per.stat$expt=='expt2'])
sd.per2 <- sd(per.stat$per.diff[per.stat$expt=='expt3'])
#combos of significant terms----------------------------------------------------
speed.sig <- mlr4[mlr4$rname == 'speed' & mlr4['Pr(>|t|)'] <bonferroni,]
speed.nsig <- mlr4[mlr4$rname == 'speed' & mlr4['Pr(>|t|)'] >=bonferroni,]
```

```
expt.sig <- mlr4[mlr4$rname == 'exptexpt3' & mlr4[,'Pr(>|t|)'] <bonferroni,]
expt.nsig <- mlr4[mlr4$rname == 'exptexpt3' & mlr4[,'Pr(>|t|)'] >=bonferroni,]

#plotting compounds with sig bias-----------------------------------------------
bias <- mlr4.predict[mlr4.predict$variable %in% unique(expt.sig$variable),]
bias$linetype <- 's'
bias$linetype[bias$variable %in% unique(speed.nsig$variable)] <- 'n'

p.bias <- ggplot(bias, aes(x=speed, y=value, colour=expt)) +
  geom_errorbar(aes(ymax=ymax, ymin=ymin)) +
  geom_point() +
    geom_line(aes(x=speed, y=prediction, linetype=linetype)) +
  facet_wrap(~variable, scales='free') +
  scale_linetype_manual(values=c(5,1), guide=FALSE) +
  scale_x_continuous(breaks=seq(0, 50000, 10000), labels=c(0:5)) +
  scale_colour_discrete(name='Experiment',labels=c('Experiment 1','Experiment 2'))
      +
  theme_bw(11) +
  xlab('Ultracentrifuge speed (10000 rpm)') +
  ylab('Concentration (mM)') +
  theme(legend.position='bottom')

ggsave('../Plots/Factor effects-mlr analysis/mlr4-bias4.png', p.bias,
       height=20, width=20, units='cm')

#Exploring biased compounds-----------------------------------------------------
bias.model <- mlr4[mlr4$variable %in% unique(expt.sig$variable),]

importance <- data.frame(variable=unique(bias.model$variable),
                         intercept=bias.model$Estimate[bias.model$rname=='(
                             Intercept)'],
                         bias=bias.model$Estimate[bias.model$rname=='exptexpt3'],
                         se=bias.model[,'Std. Error'][bias.model$rname=='exptexpt3'
                             ])
importance$impt  <- abs(importance$bias/importance$intercept*100)
importance$impt.se <- abs(importance$se/importance$intercept*100)

#exporting files----------------------------------------------------------------
write.csv(file='../Results/Bias importance.csv', importance)
write.csv(file='../Results/factor effect-expt sig.csv', expt.sig)

#Plotting compounds with significant slope (speed)------------------------------
speed <- mlr4.predict[mlr4.predict$variable %in% unique(speed.sig$variable),]

p <- ggplot(speed, aes(x=speed, y=value, colour=expt)) +
  geom_point(size=1.5) +
  geom_errorbar(aes(ymax=ymax, ymin=ymin)) +
  geom_line(aes(x=speed, y=prediction)) +
  facet_wrap(~variable, scales='free', ncol=5) +
  scale_x_continuous(breaks=seq(0, 50000, 10000), labels=c(0:5)) +
  scale_colour_discrete(name='Experiment',labels=c('Experiment 1','Experiment 2'))
      +
  theme_bw(10) +
  xlab('Ultracentrifuge speed (10000 rpm)') +
  ylab('Concentration (mM)') +
  theme(legend.position='bottom')

ggsave('../Plots/Factor effects-mlr analysis/mlr4-speed3.png', p,
       height=25, width=20, units='cm')

#export speed insig
speed.nsig <- mlr4[mlr4$rname=='speed' &mlr4[,'Pr(>|t|)'] > 0.06,]

write.csv(file='../Results/factor effect-speed nsig.csv', speed.nsig)
```

Processing method analysis.R

```
1 #////////////////////////////////////////////////////////////////////////////
```

```
#Ultracentrifuge Experiment 3 - Analysis of effects of processing methods
#ultracentrifugation vs filtration

#Pre-amble------------------------------------------------------------------------
#Setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Data source
    files/')

#Sourcing required functions
source('../Functions/function-data cleanup.R')
source('../Functions/function-descriptive statistics.R')

setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 2/R
    Scripts/Functions')
source('./ANOVA-mean.R')
source('./kruskal wallis.R')
source('./linear model.R')
source('./tukey.R')
source('./correlation.R')

#loading required packages
require(RSQLite)
require(reshape2)
require(plyr)
require(ggplot2)

#setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 3/R
    Scripts/Analysis')
#---------------------------------------------------------------------------------
#Input data
# Choose driver
drv <- dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

expt3 <-  dbGetQuery(con, "SELECT q.sampleID, q.type, q.grp, q.rep, q.class,
                     q.speed, q.process, q.filter, q.expt, d.variable, d.value
                     FROM UC3_Data d INNER JOIN Qualifier q
                     ON d.file = q.file
                     INNER JOIN Rank r
                     ON d.variable = r.Compound AND q.expt = r.expt
                     WHERE r.Confidence = 1 AND q.expt = 'expt3'
                     AND NOT (d.variable = 'Formate' AND d.value > 0.3)
                     ORDER BY q.type;")

dbDisconnect(con)

#Data clean up
    ---------------------------------------------------------------------------
clean <- data_cleanup(expt3, convert.NA=TRUE)

#Descriptive statistics-----------------------------------------------------------
stat <- descriptive.stat(clean, data.type='long', variable=c('type','variable'))

#Profile overview-----------------------------------------------------------------
plot.data <- stat
plot.data$label[stat$type %in% LETTERS[2:6]] <- 'ultracentrifuge'
plot.data$label[stat$type == 'A'] <- 'serial filter'
#plot.data$type <- factor(plot.data$type, levels=c('H','I',LETTERS[1:5]))

p <- ggplot(plot.data, aes(x=speed, y=avg, colour=label)) +
  geom_point(aes(y=value)) +
  geom_errorbar(aes(ymax=ymax, ymin=ymin)) +
  facet_wrap(~variable, scale='free') +
  scale_colour_discrete(name='Process', guide=guide_legend(reverse=TRUE)) +
  theme_bw(14) +
```

```
  xlab('Sample') +
  ylab('Concentration (mM)')

ggsave('../Plots/Processing method comparison/Profile overview.png', p,
       height=30, width=40, units='cm')

#ANOVA-------------------------------------------------------------------------

#filtration processes vs ultracentrifuge processes
anova <- ddply(stat, 'variable', aov_custom, 'class')
sig.anova <- anova[anova$p.val <= 0.05,]
write.csv(file='../Results/anova-process.csv', sig.anova)

#tukey's HSD test on filtration vs UC processes
tukey.data <- stat[stat$variable %in% sig.anova$variable,]
tukey <- ddply(tukey.data, 'variable', tukey_custom, 'type')

bonferroni <- 0.05/length(unique(tukey.data$type))
sig.tukey <- tukey[tukey$p.val <= bonferroni,]
insig.tukey <- tukey[tukey$p.val > bonferroni,]

#See which compounds do not differ significantly between
#filter-processed and low speed
diff <- sig.tukey[sig.tukey$g1 %in% c('S','X') &
                    sig.tukey$g2 %in% c('S','X'),]
same <- insig.tukey[insig.tukey$g1 %in% c('S','X') &
                      insig.tukey$g2 %in% c('S','X'),]
write.csv(file='../Results/tukey-process.csv', same)

#looking to see if filtration samples are similar to low UC speeds
low <- ddply(stat[stat$speed <= 30000,], 'variable', aov_custom, 'speed')
sig.low <- low[low$p.val <= 0.05,]
insig.low <- low[low$p.val > 0.05,]
write.csv(file='../Results/anova-process_filt vs low speed.csv', sig.low)

#linear relationships----------------------------------------------------------
#seeing if linear relationships are maintained even when including
#filtration samples --> treating filtration samples as ultrafiltration speed 0
#lumping serial filtration and single-stage filtration together, since they do not
#differ that much

lm.result <- ddply(stat, 'variable', lm_custom, 'speed')
lm.sig <- lm.result[lm.result$slope.p <= 0.05,]
lm.result$linetype <- 'ns'
lm.result$linetype[lm.result$variable %in% lm.sig$variable] <- 's'

#compounds that were linear in UC samples
uc.lm <- read.csv('../Results/correlation-speed.csv')
uc.lm <- as.character(uc.lm$variable)

#seeing if they are still linear with addition of filtration samples
#any uc.lm stop being linear?
any(uc.lm[!uc.lm %in% lm.sig$variable]) #no
#any compounds become linear after introduction of filter-processed samples?
lm.sig[!lm.sig$variable %in% uc.lm,]

corr.result <- ddply(stat, 'variable', corr_custom,
                     x='speed', y='value', methods='pearson')
corr.sig <- corr.result[corr.result$p.val <= 0.05,]
corr.insig <- corr.result[corr.result$p.val > 0.05,]
write.csv(file='../Results/correlation-process.csv', corr.sig)

#correlation of compounds that were linear in UC analysis
com.corr <- corr.sig[corr.sig$variable %in% uc.lm,]
uncom.corr <- corr.sig[!corr.sig$variable %in% uc.lm,]
write.csv(file='../Results/common correlation-process.csv', com.corr)
write.csv(file='../Results/uncommon correlation-process.csv', uncom.corr)

```

```
p.lm <- ggplot(plot.data, aes(x=speed, y=avg, colour=label)) +
  geom_abline(data=lm.result, aes(slope=slope, intercept=int,
                                  linetype=linetype),colour='grey50',) +
  geom_point(aes(y=value), position=position_dodge(width=8000)) +
  geom_errorbar(aes(ymax=ymax, ymin=ymin), position=position_dodge()) +
  scale_linetype_manual(values=c(2,1)) +
  scale_x_continuous(breaks=seq(0, 50000, 10000)) +
  facet_wrap(~variable, scale='free', ncol=7) +
  theme_bw(14) +
  theme(axis.text.x=element_text(angle=-45, hjust=0, vjust=1)) +
  xlab('Ultracentrifugation Speed (rpm)') +
  ylab('Mean Concentration (mM)')

ggsave('../Plots/Processing method comparison/Linear relationship.png', p.lm,
       height=40, width=35, units='cm')
```

Centrifugal force analysis.R

```
#////////////////////////////////////////////////////////////////////////////
#Ultracentrifuge Experiment 3 - Analysis of effects of centrifugal force
#different speeds

#Pre-amble------------------------------------------------------------------
#Setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Data source
    files/')

#Sourcing required functions
source('../Functions/function-data cleanup.R')
source('../Functions/function-descriptive statistics.R')

setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 2/R
    Scripts/Functions')
source('./ANOVA-mean.R')
source('./kruskal wallis.R')
source('./linear model.R')
source('./tukey.R')
source('./correlation.R')

#loading required packages
require(RSQLite)
require(reshape2)
require(plyr)
require(ggplot2)

#setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 3/R
    Scripts/Analysis')
#--------------------------------------------------------------------------
#Input data
# Choose driver
drv <- dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

expt3 <-  dbGetQuery(con, "SELECT q.sampleID, q.type, q.grp, q.rep, q.class,
                      q.speed, q.process, q.filter, q.expt, c.Type, m.Weight,
                      d.variable, d.value
                      FROM UC3_Data d INNER JOIN Qualifier q
                      ON d.file = q.file
                      INNER JOIN Rank r
                      ON d.variable = r.Compound AND q.expt = r.expt
                      INNER JOIN Cmpd_info c
                      ON d.variable = c.Compound
                      INNER JOIN Mol_wt m
                      ON d.variable = m.variable
                      WHERE r.Confidence = 1 AND q.expt = 'expt3'
```

```
                    AND NOT (d.variable = 'Formate' AND d.value > 0.3)
                    ORDER BY q.type;")

dbDisconnect(con)

#Data clean up-----------------------------------------------------------------
clean <- data_cleanup(expt3, convert.NA=TRUE)

#Descriptive statistics--------------------------------------------------------
stat <- descriptive.stat(clean, data.type='long', variable=c('type','variable'))

#profile overview--------------------------------------------------------------
#plotting compound levels at diff UC speeds
p.uc <- ggplot(stat, aes(x=speed, y=avg)) +
  geom_point(aes(y=value)) +
  geom_errorbar(aes(ymax=ymax, ymin=ymin)) +
  facet_wrap(~variable, scale='free') +
  theme_bw(14) +
  theme(axis.text.x=element_text(angle=-45, hjust=0, vjust=1)) +
  xlab('Ultracentrifugation Speed (rpm)') +
  ylab('Mean Concentration (mM)')

ggsave('../Plots/Speed comparison/Profile overview2.png', p.uc, height=35,width=40,
      units='cm' )

#Linear relationship-----------------------------------------------------------
#Looking for linear relationship between concentration and speed

lm.result <- ddply(stat, 'variable', lm_custom, 'speed')
lm.result$linetype <- 'ns'
lm.result$linetype[lm.result$slope.p < 0.06] <- 's'

lm.sig <- lm.result[lm.result$slope.p < 0.06,]
lm.nsig <- lm.result[!lm.result$variable %in% lm.sig$variable,]
#write.csv(file='../Results/lm non-significant.csv', lm.nsig)

#correlation
corr.result <- ddply(stat, 'variable', corr_custom,
                     x='speed', y='value', methods='pearson')
corr.sig <- corr.result[corr.result$p.val <= 0.05,]
corr.insig <- corr.result[corr.result$p.val > 0.05,]
#write.csv(file='../Results/correlation-speed2.csv', corr.sig)
#write.csv(file='../Results/weak correlation-speed2.csv', corr.insig)

#plotting lm-------------------------------------------------------------------
p.lm <- ggplot(stat, aes(x=speed, y=avg)) +
  geom_point(aes(y=value)) +
  #geom_errorbar(aes(ymax=ymax, ymin=ymin)) +
  stat_smooth(aes(y=value), method='lm', colour='grey85') +
  geom_abline(data=lm.result, aes(slope=slope, intercept=int,
                                  colour=linetype,linetype=linetype)) +
  scale_linetype_manual(values=c(2,1)) +
  scale_colour_manual(values=c('black','red')) +
  scale_x_continuous(breaks=seq(10000, 50000, 10000)) +
  facet_wrap(~variable, scale='free') +
  theme_bw(14) +
  theme(axis.text.x=element_text(angle=-45, hjust=0, vjust=1)) +
  xlab('Ultracentrifugation Speed (rpm)') +
  ylab('Mean Concentration (mM)')

ggsave('../Plots/Speed comparison/linear relationship4.png', p.lm,
       height=35, width=40, units='cm')

plot.data <- stat[stat$variable %in% unique(lm.sig$variable),]
p.siglm <- ggplot(plot.data, aes(speed, y=avg)) +
  geom_point(aes(y=value)) +
  stat_smooth(aes(y=value), method='lm', colour='black') +
  scale_x_continuous(breaks=seq(0, 50000, 10000)) +
```

```
  facet_wrap(~variable, scale='free', ncol=6) +
  theme_bw(14) +
  theme(axis.text.x=element_text(angle=-45, hjust=0, vjust=1)) +
  xlab('Ultracentrifuge Speed (rpm)') +
  ylab('Mean Concentration (mM)')

ggsave('../Plots/Speed comparison/sig linear relationship2.png', p.siglm,
       height=40, width=35, units='cm')

#Sources of variation-----------------------------------------------------------
#Kruskal wallis test
kw <- ddply(stat, 'variable', kruskal_custom, 'type')
kw.sig <- kw[kw$p.val <= 0.05,]
unique(kw$variable)

#ANOVA
anova <- ddply(stat, 'variable', aov_custom, 'type')
anova.sig <- anova[anova$p.val <= 0.05,]
write.csv(file='../Results/uc anova.csv', anova.sig)

#Post hoc, Tukey's HSD to see where difference is coming from
sig.data <- stat[stat$variable %in% anova.sig$variable,]
tukey <- ddply(sig.data, 'variable', tukey_custom, 'type')
tukey.sig <- tukey[tukey$p.val < 0.05,]

plot.data <- stat[stat$variable %in% anova.sig$variable,]
lm.anova <- lm.result[lm.result$variable %in% anova.sig$variable,]

p.anova <- ggplot(plot.data, aes(x=speed, y=avg)) +
  geom_point(aes(y=value)) +
  geom_errorbar(aes(ymax=ymax, ymin=ymin)) +
  geom_abline(data=lm.anova, aes(slope=slope, intercept=int,
                                 colour=linetype,linetype=linetype)) +
  scale_colour_manual(values='red') +
  scale_x_continuous(breaks=seq(0,50000,10000)) +
  facet_wrap(~variable, scale='free', ncol=5) +
  theme_bw(14) +
  theme(axis.text.x=element_text(angle=-45, hjust=0, vjust=1)) +
  xlab('Ultracentrifugation Speed (rpm)') +
  ylab('Mean Concentration (mM)')

ggsave('../Plots/Speed comparison/UC-anova2.png', p.anova,
       height=30,width=25, units='cm' )

#Range of changes as percent of avg---------------------------------------------
r <- stat
r <- ddply(r, 'variable', mutate, t.avg=mean(value), max=max(avg), min=min(avg))
r <- ddply(r, 'variable', mutate, p.max=(max-t.avg)/t.avg*100,
           p.min=(min-t.avg)/t.avg*100)

p.range <- unique(r[,c('variable','Type','Weight','t.avg','p.max','p.min')])
p.range$slope = NULL
p.range$int = NULL

for(i in unique(p.range$variable)) {
  p.range$slope[p.range$variable==i] <- lm.result$slope[lm.result$variable==i]
  p.range$int[p.range$variable==i] <- lm.result$int[lm.result$variable==i]
}

percent_custom <- function(d, slope='slope', int='int', x) {
  y <- d[,slope]*x+d[,int]
  per <- (y[2]-y[1])/y[1]*100

  out <- d
  out$perc.change <- per
  out$abs.change <- abs(per)
  return(out)
}
```

```
183
184 percent <- ddply(p.range,'variable', percent_custom,
185                          x=seq(10000, 50000, 10000))
186
187 #write.csv(file='../Results/Percent change per 10 000 rpm.csv', percent)
188
189 #absolute changes-------------------------------------------------------------
190 values <- unique(r[,c('speed','variable','avg')])
191
192 r2 <- ddply(values, 'variable', mutate, r.max=range(avg)[2], r.min=range(avg)[1])
193 r2 <- unique(r2[,c('variable','r.max','r.min')])
194 r2$diff <- r2$r.max-r2$r.min
195
196 #control precision------------------------------------------------------------
197 ctrl <- stat[stat$speed == 0,]
198 ctrl <- ddply(ctrl,  'variable', mutate,
199               max=max(value), min=min(value))
200 ctrl <- ddply(ctrl, 'variable', mutate, p.max=(max-avg)/avg*100,
201               p.min=(min-avg)/avg*100, p.range=(max-min)/avg*100)
202 ctrl <- unique(ctrl[,c('variable','avg','se','max','min','p.max','p.min','p.range')
    ])
203
204 #see if percent change can be categorized into -------------------------------
205 #compound class or molecular weight
206 percent.sig <- percent[percent$variable %in% lm.sig$variable,]
207
208 sig.dec <- percent.sig[percent.sig$perc.change < 0,]
209
210 percent.stat <- data.frame(avg=mean(sig.dec$abs.change),
211                            stdev=sd(sig.dec$abs.change),
212                            max=max(sig.dec$abs.change),
213                            min=min(sig.dec$abs.change))
214
215 wt.lm <- lm(data=sig.dec, formula=Weight~abs.change)
216 wt.fit <- anova(wt.lm)
217
218 wt.corr <- cor.test(sig.dec$Weight, percent$abs.change)
219
220 p.percent <- ggplot(sig.dec,aes(x=Weight, y=abs.change)) +
221   stat_smooth(method='lm')+
222   geom_point() +
223   geom_text(aes(label=variable)) +
224   theme_bw(14) +
225   xlab('Molecular weight (g/mol)') +
226   ylab('Absolute Pecent Decrease') +
227
228 ggsave('../Plots/trend by weight.png', p.percent, height=15, width=30, units='cm')
229
230 class.lm <- lm(data=percent, formula=Type~abs.change)
231
232 p.percent2 <- ggplot(sig.dec,aes(x=Type, y=abs.change)) +
233   stat_boxplot() +
234   geom_point() +
235   theme_bw(14) +
236   theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45))
```

Cross-experiment analysis.R

```
1 ////////////////////////////////////////////////////////////////////////////
    //
2 #Comparison of profiles from Expt 3, Expt 1, Expt 2 and preliminary expt
3
4 #pre-amble------------------------------------------------------------------
5 #Setting working directory
6 setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
7
8 #Sourcing required functions
9 source('function-data cleanup.R')
```

```
source('function-descriptive statistics.R')

setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 2/R Scripts/Functions')
source('./ANOVA-mean.R')
source('./kruskal wallis.R')
source('./linear model.R')
source('./tukey.R')
source('./correlation.R')

#loading required packages
require(RSQLite)
require(reshape2)
require(plyr)
require(ggplot2)

#setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 3/R Scripts/Analysis')
source('../Functions/multiple linear regression.R')
source('../Functions/mlr prediction.R')
#-----------------------------------------------------------------------------
#Input data
# Choose driver
drv <- dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

input <- dbGetQuery(con, "SELECT q.SampleID, q.type, q.grp, q.rep, q.class,
                    q.speed, q.process, q.filter, q.expt, d.variable, d.value
                    FROM UC3_Data d INNER JOIN Qualifier q
                    ON d.file = q.file
                    INNER JOIN Rank r
                    ON d.variable = r.Compound AND q.expt = r.expt
                    WHERE
                    (q.expt == 'expt1' AND q.class != 'II' AND r.Confidence = 1) OR
                    (q.expt == 'expt2' AND q.class != 'II' AND r.Confidence = 1) OR
                    q.expt IN ('expt3','preliminary') AND r.Confidence = 1
                    AND NOT (d.variable = 'Formate' AND d.value > 0.3)
                    ORDER BY q.type;")

input2 <- dbGetQuery(con, "SELECT q.SampleID, q.type, q.grp, q.rep, q.class,
                    q.speed, q.process, q.filter, q.expt, d.variable, d.value
                    FROM UC3_Data d INNER JOIN Qualifier q
                    ON d.file = q.file
                    INNER JOIN Rank r
                    ON d.variable = r.Compound AND q.expt = r.expt
                    WHERE
                    (q.expt == 'expt2' AND q.class != 'II' AND r.Confidence = 1) OR
                    q.expt = 'expt3' AND r.Confidence = 1
                    AND NOT (d.variable = 'Formate' AND d.value > 0.3)
                    ORDER BY q.type;")


dbDisconnect(con)

#Data clean up-----------------------------------------------------------------
clean <- data_cleanup(input2, convert.NA=TRUE)

#Descriptive statistics--------------------------------------------------------
stat <- ddply(clean, 'expt', descriptive.stat, data.type='long',
              variable=c('type','variable'))

#profile overview--------------------------------------------------------------
p.uc2 <- ggplot(stat, aes(x=speed, y=avg, colour=expt)) +
  geom_point(aes(y=value), alpha=0.8) +
  geom_errorbar(aes(ymax=ymax, ymin=ymin)) +
```

```
  facet_wrap(~variable, scale='free') +
  theme_bw(14) +
  theme(axis.text.x=element_text(angle=-45, hjust=0, vjust=1),
        legend.position='bottom') +
  xlab('Ultracentrifugation Speed (rpm)') +
  ylab('Mean Concentration (mM)')

ggsave('../Plots/Cross-experiment comparison/Combined expt-expt2, expt3_2.png',
       p.uc2, height=35,width=35, units='cm' )

#Removing compounds found only in one of Expt 2 and Expt 3----------------------
expt2.var <- unique(stat$variable[stat$expt=='expt2'])
expt3.var <- unique(stat$variable[stat$expt=='expt3'])

remove <- expt2.var[!expt2.var %in% expt3.var]
sub <- stat[stat$variable != remove,]

#Getting equation for multiple linear regression plane--------------------------
#checking for bias and interaction
mlr <- ddply(sub, 'variable', mlr_custom, f='value~speed+expt+speed:expt')

#compounds with insignificant slope
nsig.slope <- mlr[mlr[,'Pr(>|t|)'] > 0.05 & mlr$rname == 'speed',]

#removing compounds with insignificant slope (don't need to be plotted)
mlr.sub <- mlr[!mlr$variable %in% unique(nsig.slope$variable),]

#getting compounds with only significant bias, interaction or both
s.bias <- unique(mlr.sub$variable[mlr.sub[,'Pr(>|t|)'] < 0.06
                                  & mlr.sub$rname == 'exptexpt3'])
n.intxn <- unique(mlr.sub$variable[mlr.sub[,'Pr(>|t|)'] > 0.06
                                   & mlr.sub$rname == 'speed:exptexpt3'])
n.bias <- unique(mlr.sub$variable[mlr.sub[,'Pr(>|t|)'] > 0.06
                                  & mlr.sub$rname == 'exptexpt3'])
s.intxn <- unique(mlr.sub$variable[mlr.sub[,'Pr(>|t|)'] < 0.06
                                   & mlr.sub$rname == 'speed:exptexpt3'])

bias <- unique(mlr.sub$variable[mlr.sub$variable %in% s.bias
                                & mlr.sub$variable %in% n.intxn])
intxn <- unique(mlr.sub$variable[mlr.sub$variable %in% n.bias
                                 & mlr.sub$variable %in% s.intxn])
both <- unique(mlr.sub$variable[mlr.sub$variable %in% s.bias
                                & mlr.sub$variable %in% s.intxn])
neither <- unique(mlr.sub$variable[mlr.sub$variable %in% n.bias
                                   & mlr.sub$variable %in% n.intxn])

#Exporting compounds that followed same trend in both experiments
export.neither <- mlr.sub[mlr.sub$variable %in% neither,c(1,5,6)]
export.neither <- dcast(export.neither, variable~rname, value.var='Pr(>|t|)')
write.csv(file='../Results/mlr neither.csv', export.neither)

#predicting values along the multiple linear regressiong plane------------------
mlr.predict <- ddply(sub, 'variable', mlr_predict, f='value~speed+expt+speed:expt')

#visualizing bias and interaction-----------------------------------------------
plot.data <- mlr.predict
plot.data.bias <- plot.data[plot.data$variable %in% bias,]
plot.data.intxn <- plot.data[plot.data$variable %in% intxn,]
plot.data.both <- plot.data[plot.data$variable %in% both,]
plot.data.neither <- plot.data[plot.data$variable %in% neither,]

p <- ggplot(plot.data.both, aes(x=speed, y=value, colour=expt)) +
  geom_point() +
  geom_errorbar(aes(ymax=ymax, ymin=ymin)) +
  geom_line(aes(x=speed, y=prediction)) +
  facet_wrap(~variable, scales='free') +
  scale_x_continuous(breaks=seq(0, 50000, 10000)) +
  scale_colour_discrete(name='Experiment',labels=c('Experiment 1','Experiment 2'))
```

```
      +
144   theme_bw(11) +
145   xlab('Ultracentrifuge speed (rpm)') +
146   ylab('Concentration (mM)') +
147   theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45))
148
149 ggsave('../Plots/Cross-experiment comparison/multiple linear regression/mlr bias
      and interaction2.png', p,
150       height=10.5, width=18, units='cm')
```

Cross-experiment analysis-pca.R

```
1  #////////////////////////////////////////////////////////////////////////////
2  #Cross-experiment analysis-pca
3  #Comparison of profiles from Expt 1, Expt 2 and preliminary expt
4
5  #pre-amble-------------------------------------------------------------------
6
7  #loading requried packages
8  require(RSQLite)
9  require(ggplot2)
10
11 #sourcing required functions
12 setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions/')
13 source('function-pca.R')
14 source('function-plotting 2D score.R')
15 source('function-plotting 2D loading.R')
16 source('function-pca diagnostics.R')
17 source('function-data cleanup.R')
18 source('function-scaling.R')
19
20
21 #setting working directory
22 setwd('C:/Users/Sandi/Dropbox/Projects/Ultracentrifugation Samples/Experiment 3/R
      Scripts/Analysis')
23 #
     ----------------------------------------------------------------------------


24 #Input data
25 # Choose driver
26 drv <- dbDriver('SQLite')
27
28 # Create connection (for SQLite, this is a file)
29 con <- dbConnect(drv, '../Data source files/data.db')
30
31 input <- dbGetQuery(con, "SELECT q.sampleID, q.type, q.grp, q.rep, q.class,
32                     q.speed, q.process, q.filter, q.expt, d.variable, d.value
33                     FROM UC3_Data d INNER JOIN Qualifier q
34                     ON d.file = q.file
35                     INNER JOIN Rank r
36                     ON d.variable = r.Compound AND q.expt = r.expt
37                     WHERE
38                     (q.expt == 'expt1' AND q.class != 'II' AND r.Confidence = 1) OR
39                     (q.expt == 'expt2' AND q.class != 'II' AND r.Confidence = 1) OR
40                     q.expt IN ('expt3','preliminary') AND r.Confidence = 1
41                     AND NOT (d.variable = 'Formate' AND d.value > 0.3)
42                     ORDER BY q.type;")
43
44 input2 <- dbGetQuery(con, "SELECT q.SampleID, q.type, q.grp, q.rep, q.class,
45                     q.speed, q.process, q.filter, q.expt, d.variable, d.value
46                     FROM UC3_Data d INNER JOIN Qualifier q
47                     ON d.file = q.file
48                     INNER JOIN Rank r
49                     ON d.variable = r.Compound AND q.expt = r.expt
50                     WHERE
51                     (q.expt == 'expt2' AND q.class != 'II' AND r.Confidence = 1) OR
52                     q.expt = 'expt3' AND r.Confidence = 1
53                     AND NOT (d.variable = 'Formate' AND d.value > 0.3)
```

```
                      ORDER BY q.type;")

expt3 <- dbGetQuery(con, "SELECT q.SampleID, q.type, q.grp, q.rep, q.class,
                      q.speed, q.process, q.filter, q.expt, d.variable, d.value
                      FROM UC3_Data d INNER JOIN Qualifier q
                      ON d.file = q.file
                      INNER JOIN Rank r
                      ON d.variable = r.Compound AND q.expt = r.expt
                      WHERE
                      q.expt = 'expt3' AND r.Confidence = 1
                      AND NOT (d.variable = 'Formate' AND d.value > 0.3)
                      ORDER BY q.type;")
dbDisconnect(con)


#Data clean up-----------------------------------------------------------------
clean <- data_cleanup(expt3, convert.NA=FALSE)

#performing the PCA------------------------------------------------------------
pca.data = perform_pca(clean, sampleID='sampleID',
                       variable='variable', value='value',
                       form.y=c('sampleID','type','grp','rep','class',
                                'speed','process','filter','expt'),
                       form.x='variable',
                       scale='UV', center=TRUE, convert.NA=TRUE)

#performing diagnostics on PCA-------------------------------------------------
diagnostic.results = scree_plot(pca.data)

#unpacking the diagnostic results
diagnostic.number = diagnostic.results[['number']]
diagnostic.explained = diagnostic.results[['explained']]
diagnostic.plot = diagnostic.results[['plot']]
diagnostic.plot = diagnostic.plot + ggtitle('Scree Plot (UVN)')

#printing out diagnostic results
print(diagnostic.number)
print(diagnostic.explained)
print(diagnostic.plot)

#plotting PCA-processed data---------------------------------------------------
#Score plot 2D
#   Note: Arguments for score plot function are
#           (pca_data, qualifier_data, PCs=1:4,colour=NULL, shape=NULL, size=NULL,
#           label=NULL, title=NULL, legend.ori='horizontal', black_white=FALSE)

score_plot = score_plot_2d(pca.data, PCs=1:4,
                           colour='speed', shape='expt',
                           legend.ori='vertical',
                           black_white=FALSE)

p.score <- score_plot[['p']]
collected.data <- score_plot[['collected.data']]

p.score = p.score +
  #scale_shape_discrete(name='Process condition',
  #                     labels=c('Pre-spun, UC-processed',
  #                              'Not pre-spun, UC-processed','Filter-processed'))
      +
  scale_shape_discrete(name='Pre-process treatment', labels=c('Pre-spun','Not pre-
      spun')) +
  #scale_shape_discrete(name='Experiment', labels=c('Experiment 1','Experiment 2','
      Preliminary')) +
  guides(colour=guide_colourbar('Speed (rpm)')) +
  theme_bw(14)

ggsave('../Plots/Cross-experiment comparison/pca/score-expt3.png',p.score,
       height=10, width=25, units='cm')
```

C.3 Effects of Varying Inoculum Biomass Proportions

Data source.R

```
#////////////////////////////////////////////////////////////////////////////
#Creating database for Biomass Samples

#sourcing required functions
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
source('function-summary preprocessing.R')

#loading required packages
require(reshape2)
require(RSQLite)

#Setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Biomass samples/R Scripts/Data source files')

#preprocessing raw data
raw <- summary.preprocess('concentrations2.csv')

data <- raw[['data']]

melt.data <- melt(data)
melt.data$value <- melt.data$value/0.9

#Writing qualifiers
#reading sample names and scan order
sample.list <- read.csv('Sample list.csv')
scan <- read.csv('Scan order.csv')

#defining profiles according to scan order
qualifier <- data.frame(file=data$file, sampleID=scan$sampleID)

#re-ordering by sample ID
qualifier <- qualifier[order(qualifier$sampleID),]

#adding sample_name to qualifier
qualifier$sample_name <- sample.list$sample_name

qualifier$run <- qualifier$sample_name
qualifier$run <- gsub('.*(?<=Run )([0-9]{2}).*', '\\1', qualifier$run, perl=TRUE)

qualifier$vessel <- qualifier$sample_name
qualifier$vessel <- gsub('.*(?<=V)([0-9]).*', '\\1', qualifier$vessel, perl=TRUE)

qualifier$donor <- qualifier$sample_name
qualifier$donor <- gsub('.*(?<=, )([A-Z0-9]{2,4}).*','\\1',
                        qualifier$donor, perl=TRUE)

qualifier$day <- qualifier$sample_name
qualifier$day <- gsub('.*(?<=Day )([0-9]{1,2}).*', '\\1', qualifier$day, perl=TRUE)
qualifier$day[qualifier$day == 'Media control'] <- 0

#Compound Ranking
rank <- read.csv('Compound ranking.csv', encoding='UTF-8')
rank <- rank[,1:7]

#Compound info
#metabolite pathway
path.data <- read.csv('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/
    Data source files/Compound pathway annotation.csv',
                      header=FALSE)
path.data <- path.data[1:153,]

#Getting compound types
cmpd.type <- data.frame(Compound=t(path.data[1,2:ncol(path.data)]),
```

```
                          Type=t(path.data[2,2:ncol(path.data)]))
colnames(cmpd.type) <- c('Compound', 'Type')

# Create the connection (no file needed) ------------------------------------

# Choose driver
drv = dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con = dbConnect(drv, 'data.db')

# Creating database tables
data.table <- dbWriteTable(con,'Data', melt.data, overwrite=TRUE)

qualifier.table <- dbWriteTable(con,'Qualifier', qualifier, overwrite=TRUE)

rank.table <- dbWriteTable(con, 'Rank', rank, overwrite=TRUE)

cmpd.table <- dbWriteTable(con, 'Cmpd_info', cmpd.type, overwrite=TRUE)

tables <- dbGetQuery(con, "SELECT * FROM sqlite_master WHERE type='table';")
print(tables)

dbDisconnect(con)
```

profiling variability analysis.R

```
#////////////////////////////////////////////////////////////////////////////////////////
#Profiling variability anlaysis

setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
source('function-descriptive statistics.R')
source('function-data cleanup.R')

#loading required packages
require(RSQLite)
require(plyr)

#setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Biomass Samples/R Scripts/Analysis')

#Input data-----------------------------------------------------------------------
#Choose driver
drv <- dbDriver('SQLite')

#Create connecction (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

input <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
                          d.variable, d.value
                          FROM Data d INNER JOIN Qualifier q
                          ON d.file=q.file
                          INNER JOIN Rank r
                          ON d.variable = r.Compound
                          WHERE r.Confidence = 1
                          AND q.sampleID IN ('H1','H2','H3','R1','R2','R3')
                          ORDER BY q.sampleID;")

dbDisconnect(con)

#Data clean up-------------------------------------------------------------------
#Remove DSS, contminants and compounds of zero concentration
clean.data <- data_cleanup(input, 'sampleID', 'variable','value',
                           form.y=c('sampleID','run','vessel','donor','day'),
                           form.x='variable', contaminants=NULL, export=FALSE,
                           convert.NA=TRUE)

```

```
#Descriptive statistics-----------------------------------------------------------
#first assigning general sampleID
clean.data$set <- clean.data$sampleID
clean.data$set <- gsub('^([A-Z])(?=[0-9]).*', '\\1', clean.data$set, perl=TRUE)

stat <- ddply(clean.data, 'set', descriptive.stat, data.type='long',
              variable='variable',)

#Residuals-------------------------------------------------------------------
stat$res <- stat$value-stat$avg

#Studentized residual
stat$sres <- abs(stat$res)/stat$stdev

#plot(density(stat$sres))

#getting sum of residuals for variable
custom_residual <- function(data) {
  out <- ddply(data, 'variable', mutate, sum.res=sum(res))
  return(out)
}

stat <- ddply(stat, 'set', custom_residual)

#plot(density(stat$sum.res))

#Range-----------------------------------------------------------------------
#getting range of variable concentrations
custom_range <- function(data){
  d <- data
  out <- ddply(d, 'variable', mutate, min=min(value), max=max(value))
  out$range <- out$min-out$max
  return(out)

}
stat <- ddply(stat, 'set', custom_range)

#range as percent of mean
stat$prange <- abs(stat$range)/stat$avg*100
plot(density(stat$prange))

png(filename='../Results/Percent range2.png',
    width=10, height=7, units='cm', res=240)
plot(density(stat$prange), main='')
dev.off()

prange <- unique(stat[, c(6,8,ncol(stat))])
lrange <- prange[prange$prange >= 25,]
lrange <- lrange[order(lrange$prange),]
write.csv(file='../Results/Percent range2.csv', x=lrange)

#Relative standard deviation-------------------------------------------------
stat$rsd <- stat$stdev/stat$avg*100

plot(density(stat$rsd, bw=1))

png(filename='../Plots/precision/RSD.png',
    width=10, height=7, units='cm', res=240)
plot(density(stat$rsd, bw=1), main='')
dev.off()

high.rsd <- unique(stat[stat$rsd > 15,c('set','variable','rsd')])
write.csv(file='../Results/rsd.csv',high.rsd)
#Compounds with values greater than 2*sd-------------------------------------
stat$sd2 <- 2*stat$stdev

sd1 <- stat[abs(stat$res) > stat$stdev,]
sd2 <- stat[abs(stat$res) > stat$sd2,]
```

```
109
110 sub.stat <- unique(stat[,c(8,2:6,9:ncol(stat))])
111
112 #Precision anlaysis ANOVA-----------------------------------------------------------
113 aov_custom <- function(data, group) {
114   d <- data
115
116   grand <- mean(d$value) #grand mean
117
118   ss.grp <- ddply(d, group, mutate, avg=mean(value)) #within-group mean
119   ss.grp$grand <- grand #adding grand mean
120
121   ssr <- sum((ss.grp$value - ss.grp$avg)**2) #sum of sqr of within groups
122   ssa <- sum((ss.grp$avg - ss.grp$grand)**2) #sum of sqr of between groups
123
124   dfr <- nrow(d)-length(unique(d[,group])) #n-q
125   dfa <- length(unique(d[,group]))-1 #q-1
126
127   msa <- ssa/dfa #mean square between group
128   msr <- ssr/dfr #mean square within group
129
130   f <- msa/msr #f statistic
131   p.val <- 1-pf(f, dfa, dfr)
132   out <- data.frame(variable=unique(d$variable), f.val=f, p.val=p.val)
133   return(out)
134 }
135
136 anova <- ddply(clean.data, 'variable', aov_custom, 'set')
137
138 s.anova <- anova[anova$p.val <=0.05,]
139
140 write.csv(file='../Results/ANOVA-triplicates2.csv', s.anova)
141 #plot(density(anova$f.val))
142 p <- ggplot(stat, aes(x=set, y=value)) +
143   geom_point() +
144   stat_boxplot(aes(ymax=ymax, ymin=ymin)) +
145   facet_wrap(~variable, scales='free') +
146   theme_bw(14)
147
148 #Student's t-test-------------------------------------------------------------
149 custom_t <- function(data, group){
150   grps <- unique(data[,group])
151   g1 <- data[data[,group] == grps[1],]
152   g2 <- data[data[,group] == grps[2],]
153
154   t <- t.test(g1$value, g2$value)
155   out <- data.frame(statistic=t$statistic, p.val=t$p.value)
156   return(out)
157 }
158
159 t <- ddply(stat, 'variable', custom_t, 'set')
160 s.t <- t[t$p.val <= 0.05,]
161
162 write.csv(file='../Results/precision-t.csv', s.t)
```

duplicate variabilty analysis.R

```
1 #////////////////////////////////////////////////////////////////////////////
2 #Duplicate run variability analysis
3
4 setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
5 source('function-descriptive statistics.R')
6 source('function-data cleanup.R')
7
8 #loading required packages
9 require(RSQLite)
10 require(plyr)
11 require(ggplot2)
```

```
12
13 #setting working directory
14 setwd('C:/Users/Sandi/Dropbox/Projects/Biomass Samples/R Scripts/Analysis')
15
16 #Input data-------------------------------------------------------------------
17 #Choose driver
18 drv <- dbDriver('SQLite')
19
20 #Create connecction (for SQLite, this is a file)
21 con <- dbConnect(drv, '../Data source files/data.db')
22
23 input <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
24                           d.variable, d.value
25                           FROM Data d INNER JOIN Qualifier q
26                           ON d.file=q.file
27                           INNER JOIN Rank r
28                           ON d.variable = r.Compound
29                           WHERE r.Confidence = 1 AND q.run != 'Media control'
30                           ORDER BY q.sampleID;")
31
32 input1 <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
33                           d.variable, d.value
34                           FROM Data d INNER JOIN Qualifier q
35                           ON d.file=q.file
36                           INNER JOIN Rank r
37                           ON d.variable = r.Compound
38                           WHERE r.Confidence = 1
39                           AND q.run IN ('30', '33')
40                           ORDER BY q.sampleID;")
41
42 input2 <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
43                           d.variable, d.value
44                           FROM Data d INNER JOIN Qualifier q
45                           ON d.file=q.file
46                           INNER JOIN Rank r
47                           ON d.variable = r.Compound
48                           WHERE r.Confidence = 1
49                           AND q.run IN ('31', '32')
50                           ORDER BY q.sampleID;")
51
52 dbDisconnect(con)
53
54 #Data clean up----------------------------------------------------------------
55 #Remove DSS, contminants and compounds of zero concentration
56 clean.data <- data_cleanup(input, 'sampleID', 'variable','value',
57                            form.y=c('sampleID','run','vessel','donor','day'),
58                            form.x='variable', contaminants=NULL, export=FALSE,
59                            convert.NA=TRUE)
60
61 #Calculating mean values of triplicates in set2 and input---------------------
62 if(any(clean.data$sampleID %in% c('H1','H2','H3','R1','R2','R3'))) {
63   H.rep <- clean.data[clean.data$sampleID %in% c('H1','H2','H3'),]
64   R.rep <- clean.data[clean.data$sampleID %in% c('R1','R2','R3'),]
65
66   H <- ddply(H.rep, 'variable', mutate, avg=mean(value))
67   H$sampleID <- 'H'
68   H <- unique(H[,-7])
69   colnames(H)[7] <- 'value'
70
71   R <- ddply(R.rep, 'variable', mutate, avg=mean(value))
72   R$sampleID <- 'R'
73   R <- unique(R[,-7])
74   colnames(R)[7] <- 'value'
75
76   #removing H1,H2,H3,R1,R2,R3
77   clean.data <- clean.data[!clean.data$sampleID %in% c('H1','H2','H3','R1','R2','R3
         '),]
78   #adding mean H and R
```

```
  clean.data <- rbind(clean.data, H, R)
  print('here')
}


#casting data to give non-profiled compounds concentration of zero
data.cast <- dcast(clean.data, sampleID+run+vessel+donor+day~variable,
                   value.var='value')

data.cast[is.na(data.cast)] <- 0

#melting back to work with
data.melt <- melt(data.cast, id.vars=c('sampleID','run','vessel','donor','day'))

#adding ID for donor-run combo
data.melt$donor.run <- paste(data.melt$donor, data.melt$run, sep='.')

#Descriptive statistics----------------------------------------------------------
#stat <- descriptive.stat(data.melt, data.type='long', variable='variable')

stat <- ddply(data.melt, c('variable','donor','run'), descriptive.stat,
              data.type='long',variable='variable')

stat <- ddply(stat, c('variable','donor'), mutate, run.avg=mean(value))
stat$per.diff <- abs((stat$avg-stat$run.avg))/stat$run.avg*100

#ANOVA---------------------------------------------------------------------------
aov_custom <- function(data, group) {
  d <- data

  grand <- mean(d$value) #grand mean

  ss.grp <- ddply(d, group, mutate, avg=mean(value)) #within-group mean
  ss.grp$grand <- grand #adding grand mean

  ssr <- sum((ss.grp$value - ss.grp$avg)**2) #sum of sqr of within groups
  ssa <- sum((ss.grp$avg - ss.grp$grand)**2) #sum of sqr of between groups

  dfr <- nrow(d)-length(unique(d[,group])) #n-q
  dfa <- length(unique(d[,group]))-1 #q-1

  msa <- ssa/dfa #mean square between group
  msr <- ssr/dfr #mean square within group

  f <- msa/msr #f statistic
  p.val <- 1-pf(f, dfa, dfr)
  #out <- data.frame(variable=unique(d$variable), f.val=f, p.val=p.val)
  out <- d
  out$f.val <- f
  out$p.val <- p.val
  return(out)
}

anova <- ddply(stat, c('variable','donor'), aov_custom, group='run')
s.anova <- anova[anova$p.val <= 0.05 & !is.na(anova$p.val),]
#write.csv(file='../Results/Run comparison-31 vs 32.csv', s.anova)

#Tukey with bonferroni correction------------------------------------------------
tukey_custom <- function(data, group) {
  grps <- unique(data[,group])
  pair <- combn(grps,2)
  out <- c()
  for(i in 1:ncol(pair)) {
    grp1 <- data[data[,group] == pair[1,i],]
    grp2 <- data[data[,group] == pair[2,i],]
    grp <- rbind(grp1, grp2) #both groups being compared

    grand <- mean(data$value) #grand mean
```

```
147     ss.grp <- ddply(grp, group, mutate, avg=mean(value)) #within-group mean
148     ss.grp$grand <- grand #adding grand mean
149     ssr <- sum((ss.grp$value - ss.grp$avg)**2) #sum of sqr of within groups
150     dfr <- nrow(data)-length(unique(data[,group])) #n-q; n=# of observations,
151     #q= # of groups
152     msr <- ssr/dfr
153
154     avg1 <- mean(grp1$value)
155     avg2 <- mean(grp2$value)
156
157     n <- nrow(grp1) #number of observations in a group (this is parametric test)
158     q <- length(unique(data[,group])) #number of groups
159     se <- sqrt(msr/n)
160     q.stat <- (max(c(avg1, avg2))-min(c(avg1, avg2)))/se
161     p.val <- ptukey(q.stat, q, df=dfr, lower.tail=FALSE)
162
163     #print(pair[,i])
164     #print(grp1$value)
165     #print(grp2$value)
166     #print(msr)
167     #print(se)
168     #print(p.val)
169     #cat('\n\n')
170     output <- data.frame(g1=pair[1,i], g2=pair[2,i], p.val=p.val)
171
172     out <- rbind(out, output)
173
174   }
175   return(out)
176 }
177
178 tukey <- ddply(data.melt, 'variable', tukey_custom, group='donor.run')
179 bonferroni <- 0.05/6
180 s.tukey <- tukey[tukey$p.val <= bonferroni & !is.na(tukey$p.val),]
181
182 #t-test-----------------------------------------------------------------------
183 t_custom <- function(data, group) {
184
185   grps <- unique(data[,group])
186
187   grp1 <- data[data[,group] == grps[1],]
188   grp2 <- data[data[,group] == grps[2],]
189   grp <- rbind(grp1, grp2) #both groups being compared
190
191   result <- t.test(grp1$value, grp2$value)
192   p.val <- result$p.value
193   statistic <- result$statistic
194   output <- data.frame(g1=unique(grp1[,group]), g2=unique(grp2[,group]),
195                        stat=statistic, p.val=p.val)
196
197   return(output)
198 }
199
200 t.result <- ddply(stat,c('variable','donor'), t_custom, group='run')
201 t.sig <- t.result[t.result$p.val <= 0.05 & !is.na(t.result$p.val),]
202
203 bonferroni <- 0.05/sqrt(length(unique(stat$variable))+length(unique(stat$donor)))
204 t.sig2 <- t.result[t.result$p.val <= bonferroni & !is.na(t.result$p.val),]
205
206 for(i in c('RP','RRP','RP2','RRP2')) {
207   d1 <- unique(stat[stat$donor==i,c('variable','donor','per.diff')])
208   d2 <- t.sig[t.sig$donor == i,]
209   export <- merge(d2,d1,by='variable')
210   export <- unique(export[,c('donor.y','variable','stat','p.val','per.diff')])
211   write.csv(file=paste('../Results/run variability-',i,'.csv',sep=''),export)
212 }
213 write.csv(file='../Results/run variability-t_bonferroni.csv', t.sig2)
214
```

```
stat.sig <- stat[stat$variable %in% unique(t.sig$variable),]
stat.sig2 <- stat[stat$variable %in% unique(t.sig2$variable),]

t.list <- list(rp=unique(t.sig$variable[t.sig$donor=='RP']),
               rrp=unique(t.sig$variable[t.sig$donor=='RRP']),
               rp2=unique(t.sig$variable[t.sig$donor=='RP2']),
               rrp2=unique(t.sig$variable[t.sig$donor=='RRP2']))


#Boxplot-----------------------------------------------------------------
#significant by t test
donor <- c('RP','RRP','RP2','RRP2')
height <- c(15, 10, 8, 10)
width <- c(20,20,15,20)
n.col <- c(4,4,3,4)
for(i in 1:length(donor)){
  pbox.data <- stat[stat$donor == donor[i] & stat$variable %in% t.list[[i]],]

  p.box <- ggplot(pbox.data, aes(x=run, y=value)) +
    stat_boxplot() +
    geom_point() +
    facet_wrap(~variable, scales='free', ncol=n.col[i]) +
    theme_bw(14) +
    xlab('Run') +
    ylab('Concentration (mM)')

  #ggsave('../Plots/Run variability/anova2-sig_30 vs 33.png', p.box, height=15,
      width=25, units='cm')

  ggsave(paste('../Plots/Run variability/anova',i, '.png'), p.box,
         height=height[i], width=width[i], units='cm')
}
```

pca analysis.R

```
#////////////////////////////////////////////////////////////////////////////
#PCA Analysis of Biomass samples

#sourcing required functions
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
source('function-data cleanup.R')
source('function-pca.R')
source('function-pca diagnostics.R')
source('function-plotting 2D score.R')
source('function-plotting 2D loading.R')

#loading required packages
require(RSQLite)
require(plyr)

#setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Biomass Samples/R Scripts/Analysis')

#Input data-------------------------------------------------------------------
#Choose driver
drv <- dbDriver('SQLite')

#Create connecction (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

input <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
                          d.variable, d.value
                          FROM Data d INNER JOIN Qualifier q
                          ON d.file=q.file
                          INNER JOIN Rank r
                          ON d.variable = r.Compound
                          WHERE r.Confidence = 1
                          ORDER BY q.sampleID;")
```

```
34
35 input2 <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
36                           d.variable, d.value
37                           FROM Data d INNER JOIN Qualifier q
38                           ON d.file=q.file
39                           INNER JOIN Rank r
40                           ON d.variable = r.Compound
41                           WHERE r.Confidence = 1 AND q.run != 'Media control'
42                           ORDER BY q.sampleID;")
43
44 set1 <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
45                           d.variable, d.value
46                           FROM Data d INNER JOIN Qualifier q
47                           ON d.file=q.file
48                           INNER JOIN Rank r
49                           ON d.variable = r.Compound
50                           WHERE r.Confidence = 1 AND q.run IN (30,33)
51                           ORDER BY q.sampleID;")
52
53 set2 <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
54                           d.variable, d.value
55                           FROM Data d INNER JOIN Qualifier q
56                           ON d.file=q.file
57                           INNER JOIN Rank r
58                           ON d.variable = r.Compound
59                           WHERE r.Confidence = 1 AND q.run IN (31,32)
60                           ORDER BY q.sampleID;")
61
62 set3 <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
63                           d.variable, d.value
64                           FROM Data d INNER JOIN Qualifier q
65                           ON d.file=q.file
66                           INNER JOIN Rank r
67                           ON d.variable = r.Compound
68                           WHERE r.Confidence = 1 AND q.sampleID IN ('H1','H2','H3','
                                 R1','R2','R3')
69                           ORDER BY q.sampleID;")
70
71 dbDisconnect(con)
72
73 #Data clean up-------------------------------------------------------------------
74 #Remove DSS, contminants and compounds of zero concentration
75 clean.data <- data_cleanup(set3, 'sampleID', 'variable','value',
76                            form.y=c('sampleID','run','vessel','donor','day'),
77                            form.x='variable', contaminants=NULL, export=FALSE,
78                            convert.NA=FALSE)
79 if(FALSE){
80   export.data <- dcast(clean.data, formula=sampleID+run+vessel+donor+day~variable,
81                        value.var='value')
82   write.csv(file='../metabolite data2.csv',export.data)
83 }
84
85 #Calculating mean values of triplicates in set2 and input-----------------------
86 if(any(clean.data$sampleID %in% c('H1','H2','H3','R1','R2','R3'))) {
87   H.rep <- clean.data[clean.data$sampleID %in% c('H1','H2','H3'),]
88   R.rep <- clean.data[clean.data$sampleID %in% c('R1','R2','R3'),]
89
90   H <- ddply(H.rep, 'variable', mutate, avg=mean(value))
91   H$sampleID <- 'H'
92   H <- unique(H[,-7])
93   colnames(H)[7] <- 'value'
94
95   R <- ddply(R.rep, 'variable', mutate, avg=mean(value))
96   R$sampleID <- 'R'
97   R <- unique(R[,-7])
98   colnames(R)[7] <- 'value'
99
100  #removing H1,H2,H3,R1,R2,R3
```

```
  clean.data <- clean.data[!clean.data$sampleID %in% c('H1','H2','H3','R1','R2','R3'),]
  #adding mean H and R
  clean.data <- rbind(clean.data, H, R)
print('here')
}

#adding columns of coded run and donor labels-------------------------------------
clean.data$d <- clean.data$donor
clean.data$d <- gsub('RRP2','MET-3B', clean.data$d)
clean.data$d <- gsub('RP2', 'MET-3A', clean.data$d)
clean.data$d <- gsub('RRP', 'MET-2B', clean.data$d)
clean.data$d <- gsub('RP', 'MET-2A', clean.data$d)
clean.data$r <- clean.data$run
clean.data$r <- gsub('30',1,clean.data$r)
clean.data$r <- gsub('33',3,clean.data$r)
clean.data$r <- gsub('31',2,clean.data$r)
clean.data$r <- gsub('32',4,clean.data$r)
#Performing PCA-------------------------------------------------------------------
pca.data <- perform_pca(clean.data, sampleID='sampleID', variable='variable',
                        value='value',
                        form.y=c('sampleID','run','vessel','donor','day','r','d'),
                        form.x='variable', scale='UV', center=TRUE, convert.NA=TRUE
                            )

#Exporting pca data for Kathleen--------------------------------------------------
if(FALSE){
  pca.names <- names(pca.data)
  for(i in 1:length(pca.names)){
    write.csv(file=paste('../Results/', pca.names[i], '-triplicates.csv', sep=''),
        pca.data[[i]])
  }

  pca.summary <- as.matrix(summary(pca.data))
  write.csv(file='../Results/summary-triplicates.csv', pca.summary[[6]])
}
#performing diagnostics on PCA----------------------------------------------------
diagnostic.results = scree_plot(pca.data)

#unpacking the diagnostic results
diagnostic.number = diagnostic.results[['number']]
diagnostic.explained = diagnostic.results[['explained']]
diagnostic.plot = diagnostic.results[['plot']]
diagnostic.plot = diagnostic.plot + ggtitle('Scree Plot (UVN)')

#printing out diagnostic results
print(diagnostic.number)
print(diagnostic.explained)
print(diagnostic.plot)

#plotting PCA-processed data------------------------------------------------------
#Score plot 2D
#   Note: Arguments for score plot function are
#           (pca_data, qualifier_data, PCs=1:4,colour=NULL, shape=NULL, size=NULL,
#           label=NULL, title=NULL, legend.ori='horizontal', black_white=FALSE)

score_plot = score_plot_2d(pca.data, PCs=1:4,
                           colour='r', shape='d',label='day',
                           legend.ori='horizontal',
                           black_white=FALSE)

p.score <- score_plot[['p']]
p.score <- p.score +
  scale_colour_manual(values=c('forestgreen','darkorange2'),name='Run') + #set 3
  #scale_colour_manual(values=c('coral2','darkorange2',
  #                             'cornflowerblue','forestgreen'),name='Run') +
  scale_shape_discrete(name='Donor')
collected.data <- score_plot[['collected.data']]
```

```
166
167 ggsave('../Plots/pca plots/replicates3.png', p.score, height=15, width=25, units='
      cm')
168
169 #Loading plot 2D------------------------------------------------------------------>
170
171 #   Note: Arguments for loading plot function are (pca_data, PCs=1:4, label=NULL,
      title=NULL,
172 #                                                  view_mode='compressed')
173
174 #If you want all compound labels shown, enter all_paths in label argument
175 all_paths = rownames(pca.data$rotation)
176
177 #If you want all and only compound within the subset of observations processed in
      the PCA function, can create a temperary group called 'a' to enter in for the
      label argument
178 #a = pathway_list$Carb[pathway_list$Organic %in% rownames(pca_processed_data$
      rotation)]
179
180 loading_plot = loading_plot_2d(pca.data, PCs=1:4, label=all_paths,
181                                title='2D Loading Plot-\nUV', view_mode = '
                                       compressed')
182
183 p.load <- loading_plot[['p']]
184 load.data <- loading_plot[['collected.data']]
185
186 print(p.load)
187 ggsave('../Results/set1-loading.png', p.load, height=30, width=50, units='cm')
```

profile diversity analysis.R

```
1  #////////////////////////////////////////////////////////////////////////////
2  #Profile diversity of Biomass samples
3
4  #sourcing required functions
5  setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
6  source('function-data cleanup.R')
7  source('function-descriptive statistics.R')
8
9  #loading required packages
10 require(RSQLite)
11 require(ggplot2)
12 require(gridExtra)
13 require(RODBC)
14
15 #setting working directory
16 setwd('C:/Users/Sandi/Dropbox/Projects/Biomass Samples/R Scripts/Analysis')
17
18 #Input data-----------------------------------------------------------------------
19 #Choose driver
20 drv <- dbDriver('SQLite')
21
22 #Create connecction (for SQLite, this is a file)
23 con <- dbConnect(drv, '../Data source files/data.db')
24
25 cmpd.info <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
26                               d.variable, d.value, c.Type
27                               FROM Cmpd_info c INNER JOIN Data d
28                               ON c.Compound=d.variable
29                               INNER JOIN Qualifier q
30                               ON d.file=q.file
31                               INNER JOIN Rank r
32                               ON r.Compound=d.variable
33                               WHERE r.Confidence = 1
34                               ORDER BY q.sampleID")
35
36 dbDisconnect(con)
37
```

```
# Create connection and selecting database
con <-  odbcConnect('MySQLDSN')

input <- sqlQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
                        d.variable, d.value, t.tax_class
                        FROM biomass.data d INNER JOIN biomass.qualifier q
                        ON d.file=q.file
                        INNER JOIN general.tax t
                        ON d.variable=t.syn
                        INNER JOIN biomass.rank r
                        ON r.compound=d.variable
                        WHERE r.confidence = 1
                        ORDER by q.sampleID;")

odbcClose(con)
#data cleanup-----------------------------------------------------------------
#Remove DSS, contminants and compounds of zero concentration
clean.data <- data_cleanup(input, 'sampleID', 'variable','value',
                           form.y=c('sampleID','run','vessel','donor','day','Type')
                               ,
                           form.x='variable', contaminants=NULL, export=FALSE,
                           convert.NA=TRUE)

#Calculating mean values of triplicates in set2 and input---------------------
if(any(clean.data$sampleID %in% c('H1','H2','H3','R1','R2','R3'))) {
  H.rep <- clean.data[clean.data$sampleID %in% c('H1','H2','H3'),]
  R.rep <- clean.data[clean.data$sampleID %in% c('R1','R2','R3'),]

  H <- ddply(H.rep, 'variable', mutate, avg=mean(value))
  H$sampleID <- 'H'
  H <- unique(H[,-7])
  colnames(H)[8] <- 'value'

  R <- ddply(R.rep, 'variable', mutate, avg=mean(value))
  R$sampleID <- 'R'
  R <- unique(R[,-7])
  colnames(R)[8] <- 'value'

  #removing H1,H2,H3,R1,R2,R3
  clean.data <- clean.data[!clean.data$sampleID %in% c('H1','H2','H3','R1','R2','R3
      '),]
  #adding mean H and R
  clean.data <- rbind(clean.data, H, R)
  print('here')
}

#Adding extra qualifier to id RP-based cultures and RP2-based cultures--------
clean.data$culture.type <- 'RP'
clean.data$culture.type[clean.data$run %in% c(31,32)] <- 'RP2'
clean.data$culture.type[clean.data$run == 'Media control'] <- 'media'

#adding columns of coded run and donor labels---------------------------------
clean.data$d <- clean.data$donor
clean.data$d <- gsub('RRP2','MET-3B', clean.data$d)
clean.data$d <- gsub('RP2', 'MET-3A', clean.data$d)
clean.data$d <- gsub('RRP', 'MET-2B', clean.data$d)
clean.data$d <- gsub('RP', 'MET-2A', clean.data$d)
clean.data$r <- clean.data$run
clean.data$r <- gsub('30',1,clean.data$r)
clean.data$r <- gsub('33',3,clean.data$r)
clean.data$r <- gsub('31',2,clean.data$r)
clean.data$r <- gsub('32',4,clean.data$r)

#getting stats on each culture type
stat <- ddply(clean.data, c('variable','culture.type'), descriptive.stat,
              data.type='long',variable='variable')

#total compound concentrations------------------------------------------------
```

```
rp <- clean.data[clean.data$donor=='RP',]
rrp <- clean.data[clean.data$donor=='RRP',]
rp2 <- clean.data[clean.data$donor=='RP2',]
rrp2 <- clean.data[clean.data$donor=='RRP2',]
total <- data.frame(rp30=sum(rp$value[rp$run=='30']),
                    rp33=sum(rp$value[rp$run=='33']),
                    rrp30=sum(rrp$value[rrp$run=='30']),
                    rrp33=sum(rrp$value[rrp$run=='33']),
                    rp2.31=sum(rp2$value[rp2$run =='31']),
                    rp2.32=sum(rp2$value[rp2$run =='32']),
                    rrp2.31=sum(rrp2$value[rrp2$run =='31']),
                    rrp2.32=sum(rrp2$value[rrp2$run =='32']))
write.csv(file='../Results/total profile concentrations3.csv', total)

#Compound info - proportion of compound types-------------------------------------
cmpd.info2 <- unique(clean.data[,c('variable','donor','tax_class','d')])
cmpd.table <- table(cmpd.info2$tax_class)/5
perc.table <- cmpd.table/sum(cmpd.table)

#proportion of compound types weighted to concentration-----------------------
custom_perc <- function(data) {
  output <- ddply(data, 'sampleID', mutate, profile.total=sum(value, na.rm=TRUE))
  out <- ddply(output, c('sampleID','tax_class'), mutate,
               tax_class.total=sum(value, na.rm=TRUE))
  out <- ddply(out, 'tax_class', mutate, tax_class.avg= mean(unique(tax_class.total
      )))
  out$perc <- round(out$tax_class.total/out$profile.total*100,3)
  out <- ddply(out, 'tax_class', mutate, perc.avg=mean(unique(perc)))
  out$perc.avg <- round(out$perc.avg, 3)
  return(out)
}

cmpd.info3 <- ddply(stat[stat$sampleID != 'Y',], 'donor', custom_perc)

cmpd.info4 <- unique(cmpd.info3[,c('donor','d','tax_class','perc.avg')])
cmpd.info4$donor <- factor(cmpd.info4$donor, levels=c('RP','RRP','RP2','RRP2'))

#cast for export
cmpd.info5 <- dcast(cmpd.info4, tax_class~donor, value.var='perc.avg')
cmpd.info5 <- cmpd.info5[order(cmpd.info5$RP, decreasing=TRUE),]

#write.csv(cmpd.info5, file='../Results/profile diversity3.csv')

#looking at compounds that differ between culture types-------------------
var.carb <- as.character(unique(cmpd.info3$variable[
  cmpd.info3$tax_class == 'Carboxylic Acids and Derivatives']))

carb.acid <- unique(cmpd.info3[cmpd.info3$variable %in% var.carb,
                          c('donor','run','culture.type','variable','tax_class',
                            'avg','se','profile.total')])

var.fat <- as.character(unique(cmpd.info3$variable[
  cmpd.info3$tax_class =='Fatty Acids and Conjugates']))

fatty.acid <- unique(cmpd.info3[cmpd.info3$variable %in% var.fat,
                          c('donor','run','culture.type','variable','tax_class',
                            'avg','se','profile.total')])

#percentage of individual compounds
carb.perc <- carb.acid[order(carb.acid$variable),]

#first getting mean profile totals for each culture type
carb.perc <- ddply(carb.perc, c('variable','culture.type'), mutate,
                   avg.prof.tot = mean(profile.total))

carb.perc <- unique(carb.perc[,colnames(carb.perc) != 'profile.total'])
carb.perc$cmpd.perc <- carb.perc$avg/carb.perc$avg.prof.tot*100

```

```
fat.perc <- fatty.acid[order(fatty.acid$variable),]

#first getting mean profile totals for each culture type
fat.perc <- ddply(fat.perc, c('variable','culture.type'), mutate,
                  avg.prof.tot = mean(profile.total))

fat.perc <- unique(fat.perc[,colnames(fat.perc) != 'profile.total'])
fat.perc$cmpd.perc <- fat.perc$avg/fat.perc$avg.prof.tot*100

#visualizing proportions-----------------------------------------------------
plot.data <- unique(cmpd.info3[,c('d','tax_class','tax_class.avg','perc.avg')])
#plot.data$donor <- factor(plot.data$donor, levels=c('RP','RRP','RP2','RRP2'))
plot.data <- plot.data[order(plot.data$tax_class.avg),]
plot.data$tax_class <- factor(plot.data$tax_class, levels=unique(plot.data$tax_
    class), ordered=TRUE)

fill.val <- rev(c('coral2','dodgerblue4','seagreen','yellow2','hotpink2','brown3',
                  'cornflowerblue','darkorange2','seagreen2','purple3','tan4',
                  'bisque3','slategray'))

p <- ggplot(plot.data, aes(x=d)) +
  geom_bar(stat='identity', aes(y=tax_class.avg, fill=tax_class)) +
  scale_fill_manual(values=fill.val,
                    guide=guide_legend(reverse=TRUE,
                                       title='Compound Class')) +
  xlab('Community Culture') +
  ylab('Class Structure of Metabolite Profiles\n(weighted to concentration)') +
  theme_bw(9) +
  theme(legend.key.size=unit(0.35, 'cm'))

ggsave('../Plots/profile diversity3.png.png', p, height=10, width=15, units='cm')
```

donor comparison.R

```
#////////////////////////////////////////////////////////////////////////////
#Donor comparison

#loading required packages
require(ggplot2)
require(reshape2)
require(plyr)
require(RSQLite)

#Sourcing required functions
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions/')
source('function-data cleanup.R')
source('function-descriptive statistics.R')

setwd('C:/Users/Sandi/Dropbox/Projects/Biomass samples/R Scripts/Analysis/')

#Input data-------------------------------------------------------------------
#Choose driver
drv <- dbDriver('SQLite')

#Create connecction (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

input <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
                          d.variable, d.value
                          FROM Data d INNER JOIN Qualifier q
                          ON d.file=q.file
                          INNER JOIN Rank r
                          ON d.variable = r.Compound
                          WHERE r.Confidence = 1 AND q.run != 'Media control'
                          ORDER BY q.sampleID;")

dbDisconnect(con)

```

```
#Data clean up-----------------------------------------------------------------
#Remove DSS, contminants and compounds of zero concentration
clean.data <- data_cleanup(input, 'sampleID', 'variable','value',
                           form.y=c('sampleID','run','vessel','donor','day'),
                           form.x='variable', contaminants=NULL, export=FALSE,
                           convert.NA=TRUE)

#Calculating mean values of triplicates in set2 and input----------------------
if(any(clean.data$sampleID %in% c('H1','H2','H3','R1','R2','R3'))) {
  H.rep <- clean.data[clean.data$sampleID %in% c('H1','H2','H3'),]
  R.rep <- clean.data[clean.data$sampleID %in% c('R1','R2','R3'),]

  H <- ddply(H.rep, 'variable', mutate, avg=mean(value))
  H$sampleID <- 'H'
  H <- unique(H[,-7])
  colnames(H)[7] <- 'value'

  R <- ddply(R.rep, 'variable', mutate, avg=mean(value))
  R$sampleID <- 'R'
  R <- unique(R[,-7])
  colnames(R)[7] <- 'value'

  #removing H1,H2,H3,R1,R2,R3
  clean.data <- clean.data[!clean.data$sampleID %in% c('H1','H2','H3','R1','R2','R3
      '),]
  #adding mean H and R
  clean.data <- rbind(clean.data, H, R)
  print('here')
}

#adding columns of coded run and donor labels----------------------------------
clean.data$d <- clean.data$donor
clean.data$d <- gsub('RRP2','MET-3B', clean.data$d)
clean.data$d <- gsub('RP2', 'MET-3A', clean.data$d)
clean.data$d <- gsub('RRP', 'MET-2B', clean.data$d)
clean.data$d <- gsub('RP', 'MET-2A', clean.data$d)
clean.data$r <- clean.data$run
clean.data$r <- gsub('30',1,clean.data$r)
clean.data$r <- gsub('33',3,clean.data$r)
clean.data$r <- gsub('31',2,clean.data$r)
clean.data$r <- gsub('32',4,clean.data$r)

#casting data to give non-profiled compounds concentration of zero
data.cast <- dcast(clean.data, sampleID+run+vessel+donor+day+d+r~variable,
                   value.var='value')

data.cast[is.na(data.cast)] <- 0

#melting back to work with
data.melt <- melt(data.cast, id.vars=c('sampleID','run','vessel','donor','day','d',
    'r'))

#adding ID for donor-run combo
data.melt$donor.run <- paste(data.melt$donor, data.melt$run, sep='.')

#Descriptive statistics--------------------------------------------------------
stat <- descriptive.stat(data.melt, data.type='long', variable='variable')

#Plot of RP vs RP2-------------------------------------------------------------
plot.data <- stat
plot.data$x.code <- NULL
plot.data$x.code[plot.data$donor %in% c('RP','RRP')] <- 1
plot.data$x.code[plot.data$donor %in% c('RP2','RRP2')] <- 2
plot.data$x <- plot.data$x
plot.data$x[plot.data$run=='30'] <- plot.data$x[plot.data$run=='30'] - 0.18
plot.data$x[plot.data$run=='33'] <- plot.data$x[plot.data$run=='33'] + 0.18
plot.data$x[plot.data$run=='31'] <- plot.data$x[plot.data$run=='31'] - 0.18
plot.data$x[plot.data$run=='32'] <- plot.data$x[plot.data$run=='32'] + 0.18
```

```
101
102 #plot.data$run <- factor(plot.data$run, levels=c('30','33','31','32'))
103 #plot.data$donor <- factor(plot.data$donor, levels=c('RP','RRP','RP2','RRP2'))
104 culture <- plot.data[plot.data$donor %in% c('RP','RP2'),]
105 reciprocal <- plot.data[plot.data$donor %in% c('RRP', 'RRP2'),]
106
107
108 p <- ggplot(plot.data, aes(x=x, y=value, colour=r,linetype=d)) +
109   geom_point(aes(ymax=ymax, ymin=ymin)) +
110   stat_boxplot(data=culture,alpha=0.6) +
111   stat_boxplot(data=reciprocal, alpha=0.6) +
112   scale_x_continuous(breaks=c(1,2), label=c('MET-2','MET-3')) +
113   scale_linetype_manual(values=c(1,3,1,3), name='Defined culture') +
114   theme_bw(12) +
115   facet_wrap(~variable, scale='free_y', ncol=6) +
116   scale_colour_manual(values=c('lightcoral','chartreuse4','cyan3','darkorange'),
117                       name='Run') +
118   xlab('Defined Culture') +
119   ylab('Concentration (mM)') +
120   theme(legend.position='bottom')
121
122
123 ggsave('../Plots/donor differences4.png', p, height=25, width=23, units='cm')
```

time course analysis.R

```
1  #////////////////////////////////////////////////////////////////////////////
2  #Time-course analysis of Biomass samples
3
4  #sourcing required functions
5  setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions')
6  source('function-data cleanup.R')
7  source('../Functions/function-extract legend.R')
8  source('../Functions/function-descriptive statistics.R')
9
10 #loading required packages
11 require(RSQLite)
12 require(ggplot2)
13 require(gridExtra)
14 require(plyr)
15 require(grid)
16
17
18 #setting working directory
19 setwd('C:/Users/Sandi/Dropbox/Projects/Biomass Samples/R Scripts/Analysis')
20
21 #Input data-------------------------------------------------------------------
22 #Choose driver
23 drv <- dbDriver('SQLite')
24
25 #Create connecction (for SQLite, this is a file)
26 con <- dbConnect(drv, '../Data source files/data.db')
27
28 input <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
29                           d.variable, d.value
30                           FROM Data d INNER JOIN Qualifier q
31                           ON d.file=q.file
32                           INNER JOIN Rank r
33                           ON r.Compound=d.variable
34                           WHERE r.Confidence = 1
35                           ORDER BY q.sampleID;")
36
37 input2 <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
38                           d.variable, d.value
39                           FROM Data d INNER JOIN Qualifier q
40                           ON d.file=q.file
41                           INNER JOIN Rank r
42                           ON r.Compound=d.variable
```

```
43                              WHERE r.Confidence = 1
44                              AND q.run != 'Media control'
45                              ORDER BY q.sampleID;")
46
47 set1 <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
48                           d.variable, d.value
49                           FROM Data d INNER JOIN Qualifier q
50                           ON d.file=q.file
51                           INNER JOIN Rank r
52                           ON r.Compound=d.variable
53                           WHERE r.Confidence = 1
54                           AND q.run IN (30,33, 'Media control')
55                           ORDER BY q.sampleID;")
56
57 set2 <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
58                           d.variable, d.value
59                           FROM Data d INNER JOIN Qualifier q
60                           ON d.file=q.file
61                           INNER JOIN Rank r
62                           ON r.Compound=d.variable
63                           WHERE r.Confidence = 1
64                           AND q.run IN (31,32, 'Media control')
65                           ORDER BY q.sampleID;")
66
67 cmpd.info <- dbGetQuery(con, "SELECT q.sampleID, q.run, q.vessel, q.donor, q.day,
68                                d.variable, d.value, c.Type
69                                FROM Cmpd_info c INNER JOIN Data d
70                                ON c.Compound=d.variable
71                                INNER JOIN Qualifier q
72                                ON d.file=q.file
73                                ORDER BY q.sampleID")
74
75 dbDisconnect(con)
76
77 #data cleanup-----------------------------------------------------------------
78 #Remove DSS, contminants and compounds of zero concentration
79 clean.data <- data_cleanup(set1, 'sampleID', 'variable','value',
80                            form.y=c('sampleID','run','vessel','donor','day','Type')
                                  ,
81                            form.x='variable', contaminants=NULL, export=FALSE,
82                            convert.NA=TRUE)
83
84 #Calculating mean values of triplicates in set2 and input---------------------
85 if(any(clean.data$sampleID %in% c('H1','H2','H3','R1','R2','R3'))) {
86
87   H.rep <- clean.data[clean.data$sampleID %in% c('H1','H2','H3'),]
88   R.rep <- clean.data[clean.data$sampleID %in% c('R1','R2','R3'),]
89
90   H <- ddply(H.rep, 'variable', mutate, avg=mean(value))
91   H$sampleID <- 'H'
92   H <- unique(H[,-7])
93   colnames(H)[7] <- 'value'
94
95   R <- ddply(R.rep, 'variable', mutate, avg=mean(value))
96   R$sampleID <- 'R'
97   R <- unique(R[,-7])
98   colnames(R)[7] <- 'value'
99
100  #removing H1,H2,H3,R1,R2,R3
101  clean.data <- clean.data[!clean.data$sampleID %in% c('H1','H2','H3','R1','R2','R3
         '),]
102  #adding mean H and R
103  clean.data <- rbind(clean.data, H, R)
104
105  print('here')
106 }
107
108 #adding columns of coded run and donor labels---------------------------------
```

```
clean.data$d <- clean.data$donor
clean.data$d <- gsub('RRP2','MET-3B', clean.data$d)
clean.data$d <- gsub('RP2', 'MET-3A', clean.data$d)
clean.data$d <- gsub('RRP', 'MET-2B', clean.data$d)
clean.data$d <- gsub('RP', 'MET-2A', clean.data$d)
clean.data$r <- clean.data$run
clean.data$r <- gsub('30',1,clean.data$r)
clean.data$r <- gsub('33',3,clean.data$r)
clean.data$r <- gsub('31',2,clean.data$r)
clean.data$r <- gsub('32',4,clean.data$r)

#casting data to give non-profiled compounds concentration of zero
data.cast <- dcast(clean.data, sampleID+run+vessel+donor+day+d+r~variable,
                   value.var='value')

data.cast[is.na(data.cast)] <- 0

#melting back to work with
data.melt <- melt(data.cast, id.vars=c('sampleID','run','vessel','donor','day','d',
    'r'))

#adding ID for donor-run combo
data.melt$donor.run <- paste(data.melt$donor, data.melt$run, sep='.')

#ID for run replicates
data.melt$run.rep <- data.melt$donor.run
data.melt$run.rep[data.melt$run.rep %in% c('RP.30','RP.33','RRP.30','RRP.33')] <-
    '30.33'
data.melt$run.rep[data.melt$run.rep %in% c('RP2.31','RP2.32','RRP2.31','RRP2.32')]
    <- '31.32'

#converting day to numeric
data.melt$day <- as.numeric(data.melt$day)

#Plotting over time-bar
    ------------------------------------------------------------
plot.list <- list()
for(i in unique(data.melt$variable)) {
  plot.data <- data.melt[data.melt$variable == i,]
  plot.data$run <- factor(plot.data$run, levels=c('Media control','30','33','31','
      32'))
  plot.list[[i]] <- ggplot(plot.data, aes(x=donor, y=value, fill=day, shape=run)) +
    geom_bar(position='dodge', stat='identity') +
    facet_wrap(~ run, scales='free_x', nrow=1) +
    theme_bw(12) +
    ggtitle(i) +
    theme(legend.position='none',
          title=element_text(size=8),
          axis.text.x=element_text(hjust=0, vjust=1, angle=-45))

}

l.plot <-  ggplot(plot.data, aes(x=donor, y=value, fill=day, shape=run)) +
  geom_bar(position='dodge', stat='identity') +
  facet_wrap(~ run, scales='free_x', nrow=1) +
  theme_bw(14)
legend <- g_legend(l.plot)

png(filename='../Plots/Time-bar plots.png', width=50, height=60, units='cm',res
    =240)
do.call(grid.arrange, plot.list)
dev.off()

#Looking for linearity
    ------------------------------------------------------------
custom_linear <- function(data, group) {
  aov.lm <- dlply(data, group, aov, formula=value~day)
  out <- ldply(aov.lm, anova)
```

```
170   return(out)
171 }
172
173 lm.run <- ddply(data.melt, 'variable', custom_linear, 'run')
174 s.lm.run <- lm.run[lm.run['Pr(>F)'] <= 0.1 & !is.na(lm.run['Pr(>F)']),]
175
176 lm.dr <- ddply(data.melt, 'variable', custom_linear, 'donor.run')
177 s.lm.dr <- lm.dr[lm.dr['Pr(>F)'] <=0.1 & !is.na(lm.dr['Pr(>F)']),]
178
179 #Plotting over time-scatter
        ----------------------------------------------------------------
180
181 p.scatter <- ggplot(data.melt, aes(x=day, y=value, shape=donor, colour=run)) +
182   geom_point() +
183   geom_path() +
184   facet_wrap(~ variable, scales='free') +
185   scale_colour_manual(values=c('chartreuse4','darkorange','blue')) +
186   theme_bw(12) +
187   theme(title=element_text(size=8),
188         axis.text.x=element_text(hjust=0, vjust=1, angle=-45))
189
190 ggsave(file='../Plots/Time course/Run 31 and 32/All compounds2.png', p.scatter,
191        height=25, width=30, units='cm')
192
193 #Compounds showing linearity as defined by runs-----------------------------------
194 lm.run <- data.melt[data.melt$variable %in% unique(s.lm.run$variable),]
195
196 p.lm.run <- ggplot(lm.run, aes(x=day, y=value, shape=donor, colour=run)) +
197   geom_point() +
198   geom_path() +
199   facet_wrap(~ variable, scales='free') +
200   theme_bw(12) +
201   theme(title=element_text(size=8),
202         axis.text.x=element_text(hjust=0, vjust=1, angle=-45))
203
204 ggsave(file='../Plots/Time course/Run 31 and 32/linear-run.png', p.lm.run,
205        height=15, width=25, units='cm')
206
207 #Compounds showing linearity as defined by donors within runs----------------
208 lm.dr <- data.melt[data.melt$variable %in% unique(s.lm.dr$variable),]
209
210 p.lm.dr <- ggplot(lm.dr, aes(x=day, y=value, shape=donor, colour=run)) +
211   geom_point() +
212   geom_path() +
213   facet_wrap(~ variable, scales='free') +
214   theme_bw(12) +
215   theme(title=element_text(size=8),
216         axis.text.x=element_text(hjust=0, vjust=1, angle=-45))
217
218 ggsave(file='../Plots/Time course/Run 30 and 33/linear-donor within runs.png', p.lm
      .dr,
219        height=20, width=30, units='cm')
220
221 #Reaching common point Run 30 and 33-----------------------------------------
222 comm.lm <- c('Alanine','Asparagine','Aspartate','Betaine','Ethanol','Fructose',
223            'Glycerol','Glycolate','Hydroxyacetone','Propionate','Thymine',
224               'Uracil','Valerate')
225
226 comm.notlm <- c('Glucose','Phenylalanine','Succinate','Threonine','Tyrosine',
227                 'Xylose')
228
229 one.comm <- c('Acetate','Butyrate','Choline','Fumarate','Glycine','Isobutyrate',
230               'Isoleucine','Leucine','Lysine','Methanol','Methionine','Methylamine'
                    ,
231               'Proline','Serine','Trimethylamine','Valine')
232
233 cmpd.list <- list(comm.lm, comm.notlm, one.comm)
234
```

```
p.list <- list()
ncol <- c(4,3,4)
width <- c(17, 15, 20)
height <- c(12, 10, 20)

for(i in 1:length(cmpd.list)) {
  plot.data <- data.melt[data.melt$variable %in% cmpd.list[[i]],]
  plot.data <- plot.data[plot.data$run != 'Media control',]

  p.list[[i]] <- ggplot(plot.data, aes(x=day, y=value, shape=d, colour=r)) +
    geom_point() +
    geom_path() +
    theme_bw(11) +
    facet_wrap(~variable, scales='free', ncol=ncol[i]) +
    scale_colour_discrete(name='Run') +
    scale_shape_discrete(name='Donor') +
    scale_x_continuous(breaks=c(6,14,20)) +
    xlab('Day') +
    ylab('Concentration (mM)') +
    theme(legend.direction='horizontal',legend.position='bottom',
          legend.box='horizontal')
  }

vp <- viewport(x=0.5, y=0.5, width=unit(17,'cm'), height=unit(18,'cm'))

png(filename='../Plots/Time course/Run 30 and 33/common point2.png',
    width=17, height=25, units='cm', res=300, pointsize=11)
plot.new()
pushViewport(vp)
print(p.list[[1]],vp=viewport(x=0.5, y=0.75, width=unit(17,'cm'),
                              height=unit(16,'cm')))
print(p.list[[2]],vp=viewport(x=0.385, y=0.12, width=unit(13,'cm'),
                              height=unit(10,'cm')))
grid.text('A', x=0.03, y=1.155, gp=gpar(cex=1.5))
grid.text('B', x=0.03, y=0.36, gp=gpar(cex=1.5))
dev.off()

ggsave('../Plots/Time course/Run 30 and 33/common point-one run2.png', p.list[[3]],
       width=18, height=18, units='cm')

#Reaching common point Run 31 and 32-------------------------------------------
comm.lm <- c('Alanine','Asparagine','Glutamate','Glycine','Proline')

comm.notlm <- c('Butyrate','Glucose','Galactose','Glutamate','Hydroxyacetone',
                'Isobutyrate','Isoleucine','Isovalerate','Methionine',
                'Phenylalanine', 'Propionate','Tyrosine','Uracel','Valerate',
                'Valine','Xylose')

one.comm <- c('Acetate','Aspartate','Benzoate','Betaine','Choline', 'Ethanol',
              'Leucine','Methanol','Phenylacetate','Pyroglutamate',
              'Trimethylamine')

cmpd.list <- list(comm.lm, comm.notlm, one.comm)

p.list <- list()
ncol <- c(5,5,4)
width <- c(20, 17, 18)
height <- c(5, 16, 12)
for(i in 1:length(cmpd.list)) {
  plot.data <- data.melt[data.melt$variable %in% cmpd.list[[i]],]
  plot.data <- plot.data[plot.data$run != 'Media control',]

  p.list[[i]] <- ggplot(plot.data, aes(x=day, y=value, shape=d, colour=r)) +
    geom_point() +
    geom_path() +
    theme_bw(11) +
    facet_wrap(~variable, scales='free', ncol=ncol[i]) +
    scale_colour_manual(values=c('chartreuse4','darkorange'), name='Run') +
```

```
303     scale_shape_discrete(name='Donor') +
304     scale_x_continuous(breaks=c(6,14,20)) +
305     xlab('Day') +
306     ylab('Concentration (mM)') +
307     theme(legend.direction='horizontal',legend.position='bottom',
308           legend.box='horizontal')
309 }
310
311 vp <- viewport(x=0.5, y=0.5, width=unit(17,'cm'), height=unit(15,'cm'))
312
313 png(filename='../Plots/Time course/Run 31 and 32/common point2.png',
314     width=17, height=25, units='cm', res=300, pointsize=11)
315 plot.new()
316 pushViewport(vp)
317 print(p.list[[1]],vp=viewport(x=0.5, y=1.1, width=1, height=0.4))
318 print(p.list[[2]],vp=viewport(x=0.5, y=0.57, width=1, height=0.85))
319 grid.text('A', x=0.03, y=1.3, gp=gpar(cex=1.5))
320 grid.text('B', x=0.03, y=0.95, gp=gpar(cex=1.5))
321 dev.off()
322
323 ggsave('../Plots/Time course/Run 31 and 32/common point-one run2.png', p.list[[3]],
324        width=width[3], height=15, units='cm')
```

C.4 Metabolomic Analysis of Human Fecal Microbiota

Data source.R

```
1  #////////////////////////////////////////////////////////////////////////////
2  #This is a database of all donors and repoopulate cultures
3
4  #loading required packages
5  require(RSQLite)
6  require(reshape2)
7
8  #Setting working directory
9  setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Data source
       files/')
10
11 #Sourcing required functions
12 source('../functions/function-summary preprocessing.R')
13
14 #Getting raw data files from each project
15 #
       ------------------------------------------------------------------------------

16 #Uric acid project
17
18 rp2.ua.raw <- summary.preprocess(file.name='C:/Users/Sandi/Dropbox/Projects/Uric
       acid Project/Results/Expt 1 and 2-UA NMR/R Scripts/Data source files/UA Expt 2
       summary2.csv')
19
20 rp2.ua <- rp2.ua.raw[['data']]
21 molwt.rp2.ua <- rp2.ua.raw[['mol.wt']]
22
23 rp2.ua.melt <- melt(rp2.ua, id.vars='file')
24
25 d9.raw <- summary.preprocess('C:/Users/Sandi/Dropbox/Projects/Uric acid Project/
       Results/Expt 1 and 2-UA NMR/R Scripts/Data source files/UA Expt 1 summary2.csv'
       )
26
27 d9 <- d9.raw[['data']]
28 molwt.d9 <- d9.raw[['mol.wt']]
29
30 d9.melt <- melt(d9, id.vars='file')
```

```
#-------------------------------------------------------------------------------

#Repoopulate samples
rp1.raw <- summary.preprocess('C:/Users/Sandi/Dropbox/Projects/Repoopulate samples/
    R scripts/Data source files/Concentrations-RepoopI_mM3.csv')

rp1 <- rp1.raw[['data']]
molwt.rp1 <- rp1.raw[['mol.wt']]

rp1.melt <- melt(rp1, id.vars='file')

rp2.raw <- summary.preprocess('C:/Users/Sandi/Dropbox/Projects/Repoopulate samples/
    R scripts/Data source files/Concentrations-RepoopII_mM2.csv')

rp2 <- rp2.raw[['data']]
molwt.rp2 <- rp2.raw[['mol.wt']]

rp2.melt <- melt(rp2, id.vars='file')
#-------------------------------------------------------------------------------

#Liquid Gold samples
lg.raw <- summary.preprocess('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts
    /V 3.0/Data source files/D5_D8 summary_mM3.csv')

lg <- lg.raw[['data']]
molwt.lg <- lg.raw[['mol.wt']]


lg.melt <- melt(lg, id.vars='file')
#-------------------------------------------------------------------------------

#putting all samples together into one data frame
data <- rbind(rp2.ua.melt, d9.melt, rp1.melt, rp2.melt, lg.melt)
#-------------------------------------------------------------------------------

#getting molecular weight of compounds
cmpd.molwt <- unique(as.data.frame(t(cbind(molwt.rp2.ua, molwt.d9,
                                           molwt.rp1, molwt.rp2, molwt.lg))))
cmpd.molwt$variable <- rownames(cmpd.molwt)

#-------------------------------------------------------------------------------

#Making Qualifiers
q.rp2.ua <- data.frame(file=rp2.ua$file,
                       sampleID=LETTERS[1:17],
                       donor=rep('RP2',17),
                       vessel=c(0, rep(1,6), rep(2, 10)),
                       DPI=rep(NA, 17),
                       phase=c(0, rep(1,3), rep(2,3), rep(3,7), rep(4,3)),
                       hour=c(I(-24)
                            ,0,24,32,48,72,80,0.25,0.5,1,2,4,24,32,48,72,80),
                       dose=c(0,rep(360,3),rep(720,3),rep(360,10)),
                       dose.type=c(NA,'spike',rep('feed',15)),
                       expt=rep('UA Expt2',17))

q.rp2.ua$scan <- NA
q.rp2.ua$treatment <- c(rep('UA',14),rep('Vancomycin',3))
q.rp2.ua$run <- 5
q.rp2.ua$gen.treat <- c(rep('None', 14), rep('Vancomycin',3))
q.rp2.ua$state <- 'ss'

#making donor of class character
q.rp2.ua$donor <- as.character(q.rp2.ua$donor)
```

```
85
86 q.d9 <- data.frame(file=d9$file,
87                     sampleID=LETTERS[18:25],
88                     donor=rep('C',8),
89                     vessel=rep(3,8),
90                     DPI=c(40,41,44,45,48,49,52,53),
91                     phase=rep(NA,8),
92                     hour=rep(NA,8),
93                     dose=c(rep(0,2), rep(360,6)),
94                     dose.type=rep('feed',8),
95                     expt=rep('UA Expt1',8))
96
97 #adding columns to match other qualifier sets
98 q.d9$scan <- NA
99 q.d9$treatment <- c(rep('None',2),rep('UA',nrow(q.d9)-2))
100 q.d9$run <- 4
101 q.d9$gen.treat <- rep('None', nrow(q.d9))
102 q.d9$state <- 'ss'
103
104 #making donor of class character
105 q.d9$donor <- as.character(q.d9$donor)
106
107 q.rp1 <- data.frame(file=rp1$file,
108                     sampleID=paste('RP1.D',c(0,3,5,7,10), sep=''),
109                     DPI=c(0,3,5,7,10),
110                     donor=rep('RP1',5),
111                     expt=rep('RP1',5))
112
113 #adding columns to match other qualifier sets
114 q.rp1$vessel <- NA
115 q.rp1$phase <- NA
116 q.rp1$dose <- NA
117 q.rp1$dose.type <- NA
118 q.rp1$scan <- NA
119 q.rp1$treatment <- rep('none',nrow(q.rp1))
120 q.rp1$run <- 6
121 q.rp1$hour <- NA
122 q.rp1$gen.treat <- rep('None', nrow(q.rp1))
123 q.rp1$state <- c('uss', rep('ss',4))
124
125 #making donor of class character
126 q.rp1$donor <- as.character(q.rp1$donor)
127
128 q.rp2 <- data.frame(file=rp2$file,
129                     sampleID= paste('RP2.V',c(1,2), sep=''),
130                     DPI=rep(NA, 2),
131                     donor=rep('RP2',2),
132                     expt=rep('RP2', 2))
133
134 #adding columns to match other qualifier sets
135 q.rp2$vessel <- NA
136 q.rp2$phase <- NA
137 q.rp2$dose <- NA
138 q.rp2$dose.type <- NA
139 q.rp2$scan <- NA
140 q.rp2$treatment <- rep('None', nrow(q.rp2))
141 q.rp2$run <- 7
142 q.rp2$hour <- NA
143 q.rp2$gen.treat <- rep('None', nrow(q.rp2))
144 q.rp2$state <- 'ss'
145
146 #making donor of class character
147 q.rp2$donor <- as.character(q.rp2$donor)
148
149 q.lg <- read.csv('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Data
        source files/Liquid gold qualifier.csv',
150                  header=TRUE, check.names=FALSE, encoding='UTF-8')
151 q.lg$expt <- NA
```

```
q.lg$expt[q.lg$Donor == 'A'] <- 'Liquid gold_D5'
q.lg$expt[q.lg$Donor == 'B'] <- 'Liquid gold_D8'
q.lg <- q.lg[,colnames(q.lg) != 'Plot.Treat']
colnames(q.lg) <- c('file','scan','donor','run','treatment','DPI','state','vessel',
                    'gen.treat','sampleID','expt')

q.lg$run[q.lg$run == 1] <- 8
q.lg$run[q.lg$run == 2] <- 1
q.lg$run[q.lg$run == 3] <- 2
q.lg$run[q.lg$run == 4] <- 3

#adding columns to match other qualifier sets
q.lg$phase <- NA
q.lg$dose <- NA
q.lg$hour <- NA
q.lg$dose.type <- NA


#making donor of class character
q.lg$donor <- as.character(q.lg$donor)

#putting all qualifiers together
qualifier <- rbind(q.rp2.ua, q.d9, q.rp1, q.rp2, q.lg)
qualifier$expt <- as.character(qualifier$expt)
qualifier$vessel <- as.character(qualifier$vessel)
qualifier$state <- as.character(qualifier$state)

#-------------------------------------------------------------------------------


#Getting compound annotations
rank.rp2.ua <- read.csv('C:/Users/Sandi/Dropbox/Projects/Uric acid Project/Results/
    Expt 1 and 2-UA NMR/R Scripts/Data source files/Compound annotations-RepoopII-
    UA.csv',
                        encoding='UTF-8')
rank.rp2.ua <- rank.rp2.ua[,1:7]
rank.rp2.ua$expt <- 'UA Expt2'

rank.d9 <- read.csv('C:/Users/Sandi/Dropbox/Projects/Uric acid Project/Results/Expt
     1 and 2-UA NMR/R Scripts/Data source files/Compound annotations-Donor 9.csv',
                    encoding='UTF-8')
rank.d9 <- rank.d9[,1:7]
rank.d9$expt <- 'UA Expt1'

rank.rp1 <- read.csv('C:/Users/Sandi/Dropbox/Projects/Repoopulate samples/R scripts
    /Data source files/Compound annotations-RepoopI.csv',
                     encoding='UTF-8')
rank.rp1 <- rank.rp1[,1:7]
rank.rp1$expt <- 'RP1'

rank.rp2 <- read.csv('C:/Users/Sandi/Dropbox/Projects/Repoopulate samples/R scripts
    /Data source files/Compound annotations-RepoopII.csv',
                     encoding='UTF-8')
rank.rp2 <- rank.rp2[,1:7]
rank.rp2$expt <- 'RP2'

rank.d5 <- read.csv('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 2.0/
    data source files/Export-liquid gold confidence-D5.csv',
                    header=TRUE, encoding='UTF-8')
rank.d5 <- rank.d5[,1:7]
colnames(rank.d5) <- c('Compound','Clusters','Convolution','Consistency','
    Confidence','X..of.Profiles','X..of.Profiles')
rank.d5$expt <- 'Liquid gold_D5'

rank.d8 <- read.csv('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 2.0/
    data source files/Export-liquid gold confidence-D8.csv',
                    header=TRUE, encoding='UTF-8')
rank.d8 <- rank.d8[,1:7]
```

```
colnames(rank.d8) <- c('Compound','Clusters','Convolution','Consistency','
    Confidence','X..of.Profiles','X..of.Profiles')
rank.d8$expt <- 'Liquid gold_D8'


#putting all rankings together
rank <- rbind(rank.rp2.ua, rank.d9, rank.rp1, rank.rp2, rank.d5, rank.d8)
#
    ------------------------------------------------------------------------


#metabolite pathway
path.data <- read.csv('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/
    Data source files/Compound pathway annotation.csv',
                  header=FALSE)
path.data <- path.data[1:153,]

#Getting compound types
cmpd.type <- data.frame(Compound=t(path.data[1,2:ncol(path.data)]),
                        Type=t(path.data[2,2:ncol(path.data)]))
colnames(cmpd.type) <- c('Compound', 'Type')

#Categorizing pathways
main.path <- data.frame(main=c(rep('Carbohydrates', 12),
                               rep('Catecholamines', 5),
                               rep('Vitamins', 8),
                               rep('Antibiotics',3),
                               rep('Lipids',9),
                               rep('Amino acids', 28),
                               rep('Purine, Pyrimidine', 4),
                               rep('Fatty acids', 6),
                               rep('Organic acids', 9),
                               rep('Proteins', 5),
                               rep('Inorganic Compounds', 4),
                               rep('Steroid Hormones', 4),
                               rep('Central Metabolic Pathways', 8),
                               rep('Aromatic Compounds',8),
                               rep('Neurotransmitters',3),
                               rep('Cytochromes', 3),
                               rep('Other', 31)),
                        path=path.data[4:nrow(path.data),1])

# removing pathway headings from excel version
main <- main.path[-c(1,13,18,26,29,38,66,70,
                     76,85,90,94,98,106,114,117,120),]

#Getting pathway assignment for each compound
col.name <- as.character(as.matrix(path.data[1, 2:ncol(path.data)]))
row.name <- as.character(as.matrix(path.data[4:nrow(path.data),1]))

path <- path.data[4:nrow(path.data), 2:ncol(path.data)]
colnames(path) <- col.name
rownames(path) <- row.name
path <- path[-c(1,13,18,26,29,38,66,70,
                76,85,90,94,98,106,114,117,120),]

pathway <- c()
for(i in 1:ncol(path)) {
  temp <- path[i]
  yes <- rownames(temp)[temp[1] == 'Y']
  out <- data.frame(Compound=rep(colnames(temp), length(yes)), Pathway=yes)
  pathway <- rbind(pathway, out)
}

# Create the connection (no file needed) ------------------------------------

# Choose driver
drv = dbDriver('SQLite')

```

```
# Create connection (for SQLite, this is a file)
con = dbConnect(drv, 'data.db')

# Creating database tables
data.table <- dbWriteTable(con,'Data', data, overwrite=TRUE)

qualifier.table <- dbWriteTable(con,'Qualifier', qualifier, overwrite=TRUE)

q.rp2.ua.table <- dbWriteTable(con,'Q_RP2_UA', q.rp2.ua, overwrite=TRUE)
q.d9.table <- dbWriteTable(con,'Q_D9', q.d9, overwrite=TRUE)
q.rp1.table <- dbWriteTable(con, 'Q_RP1', q.rp1, overwrite=TRUE)
q.rp2.table <- dbWriteTable(con, 'Q_RP2', q.rp2, overwrite=TRUE)
q.lg.table <- dbWriteTable(con, 'Q_LG', q.lg, overwrite=TRUE)

rank.table <- dbWriteTable(con, 'Rank', rank, overwrite=TRUE)

molwt.table <- dbWriteTable(con, 'Mol_wt', cmpd.molwt, overwrite=TRUE)

cmpdtype.table <- dbWriteTable(con, 'Cmpd_type', cmpd.type, overwrite=TRUE)

main.table <- dbWriteTable(con, 'Main_pathway', main, overwrite=TRUE)

path.table <- dbWriteTable(con, 'Pathway', pathway, overwrite=TRUE)

tables <- dbGetQuery(con, "SELECT * FROM sqlite_master WHERE type='table';")
print(tables)

dbDisconnect(con)
```

PCA analysis.R

```
#////////////////////////////////////////////////////////////////////////////
#Modified for Manuscript A
#Analysis to compare samples across projects

#loading requried packages
require(RSQLite)
require(ggplot2)

#sourcing required functions
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions/')
source('function-pca.R')
source('function-plotting 2D score.R')
source('function-plotting 2D loading.R')
source('function-pca diagnostics.R')
source('function-data cleanup.R')
source('function-scaling.R')

#setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/LG Manuscript A')

#-----------------------------------------------------------------------
#Input data
# Choose driver
drv <- dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

input <- dbGetQuery(con, "SELECT q.sampleID, q.donor, q.run, q.vessel, q.DPI,
                    q.hour, q.phase, q.dose, q.treatment,q.gen_treat, q.expt,
                    d.variable, d.value
                    FROM Data d INNER JOIN Qualifier q
                    ON d.file = q.file
                    INNER JOIN Rank r
                    ON d.variable = r.Compound AND q.expt = r.expt
                    INNER JOIN Cmpd_type c
```

```
                        ON r.Compound = c.Compound
                        WHERE r.Confidence = 1 AND q.gen_treat = 'None'
                        AND (q.run != 8 OR q.run IS NULL)
                        AND d.variable NOT IN ('Xanthine','Glycerol')
                        AND q.state = 'ss'
                        AND q.donor != 'RP1'
                        AND q.expt != 'RP2'
                        ORDER BY q.sampleID;")

dbDisconnect(con)
#-------------------------------------------------------------------------------
#converting factors to appropriate classes
input$run <- as.character(input$run)

#-------------------------------------------------------------------------------
#Performing clearnup function to remove DSS, contaminants and compounds of
#zero concentration

clean.data <- data_cleanup(input, 'sampleID', 'variable','value',
                           form.y=c('sampleID','donor','run','vessel','DPI',
                                    'hour','phase','dose','treatment','gen_treat',
                                    'expt'),
                           form.x='variable', contaminants=NULL, export=FALSE)

clean.data$dr <- paste(clean.data$donor, clean.data$run, sep='.')
clean.data$dr <- factor(clean.data$dr, levels=c('A.1','A.2','B.3','C.4','RP2.5'))
#-------------------------------------------------------------------------------
#performing the PCA
pca.data = perform_pca(clean.data, sampleID='sampleID',
                       variable='variable', value='value',
                       form.y=c('sampleID','donor','run','vessel','DPI',
                                'hour','phase','dose','treatment','gen_treat',
                                'expt','dr'),
                       form.x='variable',
                       scale='UV', center=TRUE, convert.NA=TRUE)
#-------------------------------------------------------------------------------
#performing diagnostics on PCA
diagnostic.results = scree_plot(pca.data)

#unpacking the diagnostic results
diagnostic.number = diagnostic.results[['number']]
diagnostic.explained = diagnostic.results[['explained']]
diagnostic.plot = diagnostic.results[['plot']]
diagnostic.plot = diagnostic.plot + ggtitle('Scree Plot (UVN)')

#printing out diagnostic results
print(diagnostic.number)
print(diagnostic.explained)
print(diagnostic.plot)
#-------------------------------------------------------------------------------
#plotting PCA-processed data
#Score plot 2D
#   Note: Arguments for score plot function are
#           (pca_data, qualifier_data, PCs=1:4,colour=NULL, shape=NULL, size=NULL,
#           label=NULL, title=NULL, legend.ori='horizontal', black_white=FALSE)

score_plot = score_plot_2d(pca.data, PCs=1:4,
                           shape='dr',
                           legend.ori='horizontal',
                           outline=FALSE, black_white=FALSE)

p.score <- score_plot[['p']]
collected.data <- score_plot[['collected.data']]

ellipse_custom <- function(d) {

  #Calculating ellipse for 95% confidence
  ellipseSD = cor(d$x.value, d$y.value)
```

```
  ellipseDf = as.data.frame(ellipse(ellipseSD, scale=c(sd(d$x.value),
                                                       sd(d$y.value)),
                                    centre=colMeans(d[,c('x.value','y.value')]),
                                    t=2))
  return(ellipseDf)
}

e.path <- ddply(collected.data, 'view', ellipse_custom)


#Drawing an ellipse for 95% confidence
p.score = p.score +
  #  geom_path(data=e.path, aes(x=x, y=y), linetype=2, alpha=0.5) +
  #  scale_colour_manual(values=c('black','darkorange1','cornflowerblue',
  #                               'black','chartreuse','blue', 'grey30'),
  #                      guide_legend(title='Run'),
  #                      labels=c('NA','1','2','3','4','5','6')) +
  #scale_colour_manual(values=c('deeppink1','darkorange1','cyan4',
  #                             'darkorchid4','chartreuse','blue', 'grey30'),
  #                    guide_legend(title='Run'), labels=c(1:5)) +
  scale_shape_manual(name='', labels=c('Donor A (Run 1)','Donor A  (Run 2)',
                                       'Donor B','Donor C','MET-2'),
                     values=c(16,17,7,3,5)) +
  guides(col=guide_legend(byrow=TRUE))

ggsave('../Plots/PCA plots/Profiles-score5.png', p.score, height=15, width=25,
    units='cm')

tiff(file='C:/Users/Sandi/Dropbox/Projects/Liquid Gold/Manuscript/Manuscript A/
    Journal of Proteome Research/Figure 2.tif',
          width=7, height=10/2.54, units='in', res=300,)
print(p.score)
dev.off()
#
    ------------------------------------------------------------------------------------------------------------------------>


#Loading plot 2D
#   Note: Arguments for loading plot function are (pca_data, PCs=1:4, label=NULL,
    title=NULL,
#                                                  view_mode='compressed')

#If you want all compound labels shown, enter all_paths in label argument
all_paths = rownames(pca.data$rotation)

source('./Functions/defined binning.R')
bin <- defined_binning()

#If you want all and only compound within the subset of observations processed in
    the PCA function, can create a temperary group called 'a' to enter in for the
    label argument
#a = pathway_list$Carb[pathway_list$Organic %in% rownames(pca_processed_data$
    rotation)]
loading_plot = loading_plot_2d(pca.data, PCs=1:4, label=all_paths,
                               title=NULL, outline=FALSE, view_mode = 'compressed')

p.load <- loading_plot[['p']]
load.data <- loading_plot[['collected.data']]

load.data$bin <- NA
for(i in bin$variable) {
  load.data$bin[load.data$Text == i] <- as.character(bin$bin[bin$variable == i])
}

p.load <- p.load +
  geom_point(data=load.data, aes(x=xPC, y=yPC, shape=bin), size=1) +
  geom_text(data=load.data, aes(x=xPC, y=yPC, label=Text),
            hjust=-0.1, vjust=0, size=1, colour='grey50') +
  scale_colour_manual(values=c('deeppink1','darkorange1','cyan4',
```

```
                                'darkorchid4','chartreuse'),
                        guide_legend(title='Bin')) +
  scale_shape_manual(name='Bin',values=rev(c(16,17,7,3,5))) +
  theme(legend.position='bottom')

ggsave('../Plots/PCA plots/Profiles-loading5.png', p.load,
       height=12*1.5, width=18*2, units='cm')

#EPS-------------------------------------------------
p <- ggplot(load.data, aes(xPC, yPC, shape=bin)) +
  geom_text(aes(label=Text),size=4,hjust=-0.1, vjust=0.4, colour='grey50') +
  geom_point(size=3) +
  scale_shape_manual(name='Bin', values=rev(c(16,17,7,3,5))) +
  facet_wrap(~ View, ncol=min(floor(length(1:4)/2), 3)) +
  xlab('PC score') +
  ylab('PC score') +
  theme_bw(12) +
  theme(axis.title.x = element_text(size=14, face='bold'),
        axis.title.y = element_text(size=14,face='bold'),
        #axis.text = element_text(size=12, colour='black'),
        legend.title = element_text(size=14),
        legend.text = element_text(size=14),
        legend.key = element_rect(colour='white'),
        legend.margin = unit(0, "cm"),
        panel.border=element_rect(size=2, colour='black'),
        strip.text.x = element_text(size=12, face='bold'),
        legend.position='bottom')
#Drawing origin
p = p + geom_vline(xintercept=0, alpha=0.3)
p = p + geom_hline(yintercept=0, alpha=0.3)

#Calculating plot limits
xlimit = max(abs(load.data$xPC))*1.15
ylimit = max(abs(load.data$yPC))*1.1

#Setting limits
p = p + xlim(-xlimit,xlimit)
p = p + ylim(-ylimit,ylimit)


ggsave('../Plots/PCA plots/temp.png', p, height)

tiff(filename='C:/Users/Sandi/Dropbox/Projects/Liquid Gold/Manuscript/Manuscript A/
    Journal of Proteome Research/Figure S2.tif',
       width=15*2/2.54, height=12*2/2.54, units='in',res=300,
       pointsize=10)
grid.arrange(p.score, p, nrow=2)
grid.text('A', x=unit(0.7,'cm'), y=unit(23.25, 'cm'),gp=gpar(cex=2))
grid.text('B', x=unit(0.7,'cm'), y=unit(11.3, 'cm'),gp=gpar(cex=2))
dev.off()
```

Profile diveristy analysis.R

```
#////////////////////////////////////////////////////////////////////////////
#Modified for Liquid Gold Manuscript A
#Profile diversity

#Setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Analysis')

#loading required packages
require(RSQLite)
require(ggplot2)
require(plyr)
require(reshape2)
require(gridExtra)

#sourcing required functions
```

```
source('../Functions/function-data cleanup.R')
source('../Functions/function-descriptive statistics.R')
source('../Functions/function-scaling.R')
source('../Functions/function-extract legend.R')
source('../Functions/function-multiplot.R')
source('../Functions/function-heat map bin.R')
source('../Functions/function-defined binning.R')
source('../Functions/function-siegel tukey test.R')
source('../Functions/function-weighted compound class.R')


#setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/LG Manuscript A')

#Input data
------------------------------------------------------------------------------
# Choose driver
drv <- dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

input <- dbGetQuery(con, "SELECT q.sampleID, q.donor, q.run, q.vessel, q.DPI,
                    q.hour, q.phase, q.dose, q.treatment,q.gen_treat, q.expt,
                    d.variable, d.value, c.Type
                    FROM Data d INNER JOIN Qualifier q
                    ON d.file = q.file
                    INNER JOIN Rank r
                    ON d.variable = r.Compound AND q.expt = r.expt
                    INNER JOIN Cmpd_type c
                    ON r.Compound = c.Compound
                    WHERE r.Confidence = 1 AND q.gen_treat = 'None'
                    AND (q.run != 8 OR q.run IS NULL)
                    AND d.variable NOT IN ('Xanthine','Glycerol')
                    AND q.state = 'ss'
                    AND q.donor != 'RP1'
                    AND q.expt != 'RP2'
                    ORDER BY q.sampleID;")

cmpd.info <- dbGetQuery(con, "SELECT c.Compound, c.Type, p.Pathway, m.main
                        FROM Cmpd_type c INNER JOIN Pathway p
                        ON c.Compound = p.Compound
                        INNER JOIN Main_pathway m
                        ON p.Pathway = m.path
                        ORDER BY c.Compound")

dbDisconnect(con)

#data cleanup
-------------------------------------------------------------------------------

#Performing data cleanup to remove zero value compounds
#also casts data into wide format
clean <- data_cleanup(input, convert.NA=TRUE)


#Ordering compound classes by abundance-----------------------------------------
rp <- input[input$donor =='RP2',]
d <- input[input$donor != 'RP2',]

div.data <- clean
div.table <- as.data.frame(table(div.data$Type))
div.table <- div.table[order(div.table$Freq, decreasing=TRUE),]
var1.levels <- as.character(div.table$Var1)
div.data$Type <- factor(div.data$Type, levels=var1.levels, ordered=TRUE)

#Plotting profile diversity --> what times of compounds are the profiles made of
```

```
p <- ggplot(div.data, aes(x=donor, fill=Type)) +
  geom_bar(stat='bin') +
  theme_bw(14)
ggsave('./Plots/Profile diversity/profile diversity_order2.png', p, height=20,
    width=15, units='cm')

#Abundance of compound scaled to concentrations-------------------------------------
div <- ddply(div.data, 'donor', custom_perc,'Type')

p.div <- ggplot(div, aes(x=donor, y=class.avg, fill=Type)) +
  geom_bar(stat='identity') +
  theme_bw(14)

#export compount proportions
div.cast <- dcast(div, formula=Type~donor, value.var='perc')
write.csv(file='./Results/Profile diversity-weighted.csv',div.cast)

#Percentage of profiles is made of each compound class
    ---------------------------------
#(not scaled to concentration)
custom_table <- function(data){
  table <- as.data.frame(table(data$Type))
  prop <- table$Freq/sum(table$Freq)*100
  type <- data.frame(donor=unique(data$donor),type=table$Var1, prop)
  return(type)
}
table <- ddply(clean, 'donor', custom_table)
type.data <- dcast(table, donor ~ type, value.var='prop' )
write.csv(file='./Results/profile diversity2.csv', x=type.data)
```

Profile analysis.R

```
#////////////////////////////////////////////////////////////////////////////////
    /////////
#Modified for Manuscript A
#Cross-experiment analysis

#Setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Analysis')

#loading required packages
require(RSQLite)
require(ggplot2)
require(plyr)
require(reshape2)
require(gridExtra)
require(grid)
require(gridBase)

#sourcing required functions
source('../Functions/function-data cleanup.R')
source('../Functions/function-descriptive statistics.R')
source('../Functions/function-scaling.R')
source('../Functions/function-extract legend.R')
source('../Functions/function-multiplot.R')
source('../Functions/function-heat map bin.R')


#Input data
    ------------------------------------------------------------------------------
# Choose driver
drv <- dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

input <- dbGetQuery(con, "SELECT q.sampleID, q.donor, q.run, q.vessel, q.DPI,
                    q.hour, q.phase, q.dose, q.treatment,q.gen_treat,
```

```
                        q.expt, d.variable, d.value
                        FROM Data d INNER JOIN Qualifier q
                        ON d.file = q.file
                        INNER JOIN Rank r
                        ON d.variable = r.Compound AND q.expt = r.expt
                        WHERE r.Confidence = 1 AND q.gen_treat = 'None'
                        AND d.variable NOT IN ('Xanthine','Glycerol')
                        AND q.state = 'ss'
                        AND (q.run != 8 OR q.run IS NULL)
                        AND q.donor != 'RP1'
                        AND q.expt != 'RP2'
                        ORDER BY q.sampleID;")

cmpd.info <- dbGetQuery(con, "SELECT c.Compound, c.Type, p.Pathway, m.main
                            FROM Cmpd_type c INNER JOIN Pathway p
                            ON c.Compound = p.Compound
                            INNER JOIN Main_pathway m
                            ON p.Pathway = m.path
                            ORDER BY c.Compound")

dbDisconnect(con)

#Setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/LG Manuscript A/
    Analysis')

#data cleanup
    ----------------------------------------------------------------------------

#Performing data cleanup to remove zero value compounds
#also casts data into wide format
clean <- data_cleanup(input, form.y=c('sampleID','donor','run','vessel',
                                      'DPI','hour','phase','dose','treatment',
                                      'gen_treat', 'expt'),
                      form.x='variable', convert.NA=TRUE)

#make sure all compounds have a value for each donor
data.cast <- dcast(clean, formula=sampleID+donor+run+vessel+DPI+hour+phase+dose+
                       treatment+gen_treat+expt~variable, value.var='value')

data.cast[is.na(data.cast)] <- 0

#melting back to long format
data.melt <- melt(data.cast, id.vars=c('sampleID','donor','run','vessel',
                                       'DPI','hour','phase','dose','treatment',
                                       'gen_treat', 'expt'))

#separating out qualifier and data
qualifier <- data.melt[,1:11]
x.data <- data.melt[,12:13]

#descriptive stats
    ----------------------------------------------------------------------------
#Getting descriptive stats for each donor
stat <- ddply(data.melt, 'donor', descriptive.stat, data.type='long', variable='
    variable',
              qualifier=qualifier, scale.data=FALSE)
#write.csv(file='check stat.csv', stat)

#density plot
    ----------------------------------------------------------------------------
#getting kernal density plot of values
p.dens <- ggplot(stat[stat$avg >= 0.1 & stat$avg <=20,]) +
  geom_density(aes(x=avg)) +
  theme_bw(16)

plot(density(stat$avg))

```

```
plot(density(stat$avg))
png(file='../Plots/bin density.png', height=10, width=15, units='cm', res=240)
plot(density(stat$avg), main='', xlab='Mean Concentration (mM)',
     sub='bw=0.03089')
dev.off()

temp <- stat$avg[stat$avg >=0.1 & stat$avg <=20]
plot(density(temp)) #bw=0.135

png(file='../Plots/bin density-inset.png', height=7.5, width=10, units='cm', res
    =240)
plot(density(stat$avg[stat$avg >=0.1 & stat$avg <=20]),
     main='', xlab='', ylab='', cex=10, xaxt='n')
axis(1, at=seq(0,20,5), labels=c(0.1, seq(5,20,5)))
dev.off()

vp <- viewport(x=0.55, y=0.3, width=0.95, height=0.95)
vp.inset <- viewport(x=0.55, y=0.8, width=unit(12, 'cm'),
                     height=unit(8.5, 'cm'))
tiff(filename='C:/Users/Sandi/Dropbox/Projects/Liquid Gold/Manuscript/Manuscript A/
    Journal of Proteome Research/Figure S1.tif',
         width=7, height=5.5, units='in', res=300,
         pointsize=10)
plot.new()
pushViewport(vp)
plot(density(stat$avg), main='', xlab='Mean Concentration (mM)',
     sub='bw=0.03089')
pushViewport(vp.inset)
par(new=TRUE, fig=gridFIG())
plot(density(stat$avg[stat$avg >=0.1 & stat$avg <=20]),
     main='', xlab='', ylab='', cex=10, xaxt='n')
axis(1, at=seq(0,20,5), labels=c(0.1, seq(5,20,5)))
upViewport()
grid.lines(x=c(0.85,0.26), y=c(0.64,0.375))
grid.lines(x=c(0.298,0.062), y=c(0.64,0.45))
dev.off()

#Binning
-------------------------------------------------------------------------------
#Assigning data to bins
bin.data <- bin.heatmap(stat, bin=c(20,10,1,0.1,0), 'variable',
                        order.data=stat, order.var='variable', order.value='avg')

uni.list <- bin.data[['uni.list']]
ordered.data <- bin.data[['ordered.data']]
ordered.data$donor <- factor(ordered.data$donor, levels=c('RP2','A','B','C'))

#Padded variable names
ordered.data$pad <- sprintf('%022s', as.character(ordered.data$variable))

pad.list <- list()
for(i in 1:length(uni.list)) {
  pad.temp <- sprintf('%022s', as.character(uni.list[[i]]))
  pad.list[[i]] <- pad.temp
}

pad.order <- c()
for(i in 1:length(pad.list)) {
  temp <- as.character(pad.list[[i]])
  pad.order <- c(pad.order, temp)
}


#plotting heat maps
-------------------------------------------------------------------------------
#plotting heat maps by bin
plot.list <- list()
legend.list <- list()
```

```
fig.list <- list()
fig.leg <- list()


fig.width <- c(7.7, 7.5, 7.8, 8.8, 9.4)
fig.height <- c(2.65,3.1,4.7,13.5,10.9)
margin <- list(c(0,0.2,0.5,0.8),
               c(0,0.2,0.5,0.8),
               c(0,0.2,0.3,0.4),
               c(0,0.2,0,0),
               c(0,0.2,0,0))

for(i in 1:length(uni.list)) {
  plot.data <- ordered.data[ordered.data$variable %in% uni.list[[i]],]

  #setting compound order
  plot.data$pad <- factor(as.character(plot.data$pad),
                          levels=rev(pad.order[pad.order %in% pad.list[[i]]]),
                              ordered=TRUE)

  plot.list[[i]] <- ggplot(plot.data, aes(x=donor, y=pad)) +
    geom_tile(aes(fill=avg), colour='grey90') +
    scale_fill_gradient(low='grey90', high='grey0',
                        name='Mean\nConcentration (mM)') +
    scale_x_discrete(labels=c('MET-2','A','B','C')) +
    theme_bw(10) +
    xlab('Donor') +
    theme(legend.position=c(0.52, 0.5), legend.direction='horizontal',
          legend.key.width=unit(1, 'cm'),
          legend.key.height=unit(0.3,'cm'),
          legend.text=element_text(size=7.5),
          legend.title=element_text(size=7.5),
          axis.title.y=element_blank(),
          plot.margin=unit(margin[[i]], 'cm'))


  legend.list[[i]] <- g_legend(plot.list[[i]])

  fig.list[[i]] <- grid.arrange(legend.list[[i]],
                                plot.list[[i]] + theme(legend.position='none'),
                                heights=c(0.1, 0.9))

  #save as panels
  if(FALSE){
    png(filename=paste('../Plots/Profiles/profile4_bin',i,'.png', sep=''),
        height=fig.height[i], width=fig.width[i], units='cm', res=240, pointsize
            =10)
    fig.list[[i]] <- grid.arrange(plot.list[[i]] + theme(legend.position='none'))
    dev.off()

    png(filename=paste('../Plots/Profiles/profile4_bin_leg',i,'.png', sep=''),
        height=1, width=leg.width[i], units='cm', res=240, pointsize=10)
    fig.leg[[i]] <- grid.arrange(legend.list[[i]])
    dev.off()
  }

}

#Setting plot dimensions
    ----------------------------------------------------------------------

w <- list(c(10,7.7),     #1
          c(10,7.5),     #2
          c(10,7.8),     #3
          c(10,8.8),     #4
          c(10,9.4))     #5

h <- list(c(1,2.5),     #1
```

```
227             c(1,2.9),     #2
228             c(1,4.5),     #3
229             c(1,13.5),    #4
230             c(1,10.3))    #5
231
232 #EPS-------------------------------------------------------
233 tiff(filename='C:/Users/Sandi/Dropbox/Projects/Liquid Gold/Manuscript/Manuscript A/
        Journal of Proteome Research/Figure 3.tif',
234     width=7, height=sum(c(h[[4]],h[[3]])/2.54), units='in', res=300,
235     pointsize=11)
236 grid.arrange(arrangeGrob(legend.list[[1]],
237                          plot.list[[1]] + theme(legend.position='none'),
238                          legend.list[[2]],
239                          plot.list[[2]] + theme(legend.position='none'),
240                          legend.list[[5]],
241                          plot.list[[5]] + theme(legend.position='none'),
242                          ncol=1, heights=c(h[[1]],h[[2]],h[[5]]),
243                          widths=c(w[[1]],w[[2]],w[[5]])),
244              arrangeGrob(legend.list[[3]],
245                          plot.list[[3]] + theme(legend.position='none'),
246                          legend.list[[4]],
247                          plot.list[[4]] + theme(legend.position='none'),
248                          ncol=1, heights=c(h[[3]],h[[4]]),
249                          widths=c(w[[3]],w[[4]])),
250              ncol=2)
251 grid.text('A', x=unit(0.25,'cm'), y=unit(19.45, 'cm'),gp=gpar(cex=1.5))
252 grid.text('B', x=unit(0.25,'cm'), y=unit(15.7, 'cm'),gp=gpar(cex=1.5))
253 grid.text('E', x=unit(0.25,'cm'), y=unit(11.52, 'cm'),gp=gpar(cex=1.5))
254 grid.text('C', x=unit(9.1,'cm'), y=unit(19.45, 'cm'),gp=gpar(cex=1.5))
255 grid.text('D', x=unit(9.1,'cm'), y=unit(14, 'cm'),gp=gpar(cex=1.5))
256 dev.off()
257
258 #Mann Whitney test
        ---------------------------------------------------------------------------
259 #seeing what compounds differ b/w donor and RPs
260 mw_custom <- function(data, value='value') {
261
262   d <- data[data$donor %in% c('A','B','C'),]
263   rp <- data[data$donor == 'RP2',]
264   out <- c()
265   test.result=wilcox.test(d[,value], rp[,value])
266
267   output <- rp
268
269   output$p.val <- test.result$p.value
270   output$stat <- test.result$statistic
271
272   out <- rbind(out, output)
273   return(out)
274 }
275
276
277 mw.result <- ddply(stat, 'variable ', mw_custom)
278
279 sig <- mw.result[mw.result$p.val <= 0.05,]
280
281
282 sig.bin <- uni.list
283
284 #Putting significant compounds in bin order-----------------------------------
285 sig.order <- list()
286 for(i in 1:length(sig.bin)) {
287   sig.data <- sig[sig$variable %in% sig.bin[[i]],]
288   sig.data <- unique(sig.data[,c('variable','p.val')])
289
290   type <- cmpd.info[cmpd.info$Compound %in% unique(sig.data$variable),]
291   for(j in unique(type$Compound)){
292     type$p.val[type$Compound == j] <- sig.data$p.val[sig.data$variable == j]
```

```
293       #print('end1')
294     }
295
296     #type$Compound <- factor(type$Compound,
297     #                        levels=rev(levels(sig.bin[[i]])),order=TRUE)
298     #type <- type[order(type$Compound),]
299     type <- type[order(type$p.val, decreasing=FALSE),]
300
301     sig.order[[i]] <-type
302     #print('end2')
303
304 }
305
306 #Putting all in one dataframe -- Compound Type
307 sig.all.cmpd <- do.call(rbind.data.frame, sig.order)
308 sig.all.cmpd <- unique(sig.all.cmpd[,c(1,5,2)])
309 table(sig.all.cmpd$Type)
310
311 #Putting all in one dataframe -- Pathway
312 sig.all.path <- do.call(rbind.data.frame, sig.order)
313 sig.all.path <- unique(sig.all.path[,c(1,5,3)])
314 table(sig.all.path$Pathway)
315
316 #write.csv(file='../Results/donor-pvalue2.csv', sig.all)
317
318 #exporting each bin separately
319 for(i in 1:length(sig.order)) {
320   sig.write <- sig.order[[i]]
321   #write.csv(file=paste('../Results/donor-pvalue',i,'_2.csv',sep=''),sig.write)
322 }
323
324 #Seeing how these compounds differ across donors-----------------------------
325 d <- sig.all
326
327 out <- data.frame(A=NULL, B=NULL, C=NULL, RP2=NULL)
328 for(i in d$Compound){
329   avg <- unique(stat[stat$variable == i, c('donor','avg')])
330   tmp <- as.data.frame(t(avg))
331   val <- tmp[2,]
332   colnames(val) <- as.character(as.matrix(tmp[1,]))
333   val$variable <- i
334   out <- rbind(out, val)
335 }
336
337 d <- cbind(d, out)
338
339 #Getting mean concentrations of each group-----------------------------------
340 a <- stat[stat$variable %in% sig.all$Compound,]
341
342 rp <- a[a$donor == 'RP2',]
343 dn <- a[a$donor %in% c('A','B','C'),]
344
345 rp$donor.type <- 'rp'
346 dn$donor.type <- 'd'
347
348 rp <- ddply(rp, 'variable', mutate, rp.avg=mean(unique(avg)))
349 dn <- ddply(dn, 'variable', mutate, dn.avg=mean(unique(avg)))
350
351 rp2 <- unique(rp[,c('donor.type','variable','rp.avg')])
352 dn2 <- unique(dn[,c('donor.type','variable','dn.avg')])
353
354 b <- cbind(rp2[c('variable','rp.avg')], 'dn.avg'=dn2$dn.avg)
355
356 source('../Functions/defined binning.R')
357 bin.info <- defined_binning()
358 b$bin <- NULL
359 for(i in b$variable)b$bin[b$variable == i] <- bin.info$bin[bin.info$variable==i]
360
```

```
h.rp <- b[b$rp.avg > b$dn.avg,]
l.rp <- b[b$rp.avg < b$dn.avg,]

#Visualize in plot form-----------------------------------------------------------
d.melt <- melt(d, id.vars=c('Compound','variable','Type','p.val'),
                    measure.vars=c('A','B','C','RP2'), variable.name='donor')
d.melt$value <- as.numeric(d.melt$value)

plot.var <- unique(d.melt$variable)
plot.var <- factor(plot.var, levels=unlist(uni.list))
plot.var <- plot.var[order(plot.var)]
plot.var <- as.character(plot.var)

plot.function <- function(.x, dat, var, variable) {
  plot.data <- dat[dat[,variable]==var[.x],]
  p <- ggplot(plot.data, aes(x=donor, y=variable))  +
    geom_tile(aes(fill=value)) +
    geom_text(aes(fill=value, label=round(plot.data$value,3))) +
    scale_fill_gradient(low='white', high='blue') +
    theme_bw(14) +
    theme(axis.title.y=element_blank())
  return(p)
}
p <- lapply(1:length(plot.var), plot.function,
            dat=d.melt, var=plot.var, variable='variable')

png('../Plots/test.png', height=100, width=15, units='cm', res=240)
do.call(grid.arrange, c(p, ncol=1))
dev.off()
#pathway analysis-----------------------------------------------------------------
path.table <- as.data.frame(table(data$Pathway))
path.table <- path.table[order(path.table$Freq, decreasing=TRUE),]
#write.csv(file='../Results/donor-pathway.csv', path.table)

main.table <- as.data.frame(table(data$main))
main.table <- main.table[order(main.table$Freq, decreasing=TRUE),]
#write.csv(file='../Results/donor-main pathway.csv', main.table)

data$main.count <- NA
for(i in unique(main.table$Var1)) {
  data$main.count[data$main == i] <- main.table$Freq[main.table$Var1 == i]
}

data$path.count <- NA
for(i in unique(path.table$Var1)){
  data$path.count[data$Pathway == i] <- path.table$Freq[path.table$Var1 == i]
}

data <- data[order(-data$path.count),]
path.level <- unique(data$Pathway)
data$Pathway <- factor(data$Pathway, levels=rev(path.level), ordered=TRUE)
cmpd.level <- unique(data$Compound)
data$Compound <- factor(data$Compound, levels=cmpd.level, ordered=TRUE)

p.path <- ggplot(data[data$main %in% c('Amino acids', 'Organic acids',
                                       'Vitamins','Central Metabolic Pathways',
                                       'Lipids','Purine, Pyrimidine','Fatty acids')
                                          ,],
                 aes(x=main,fill=Pathway)) +
  geom_bar(stat='bin', colour='black') +
  theme_bw(14) +
  theme(axis.text.x=element_text(hjust=0, vjust=1, angle=-45))
```

Biomarker analysis.R

```
#////////////////////////////////////////////////////////////////////////////
#Analysis for biomarkers unique to each donor
#Created for Liquid Gold Manuscript A
```

```
#(not in regular Liquid Gold manuscript R analyses)

#Setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Analysis')

#loading required packages
require(RSQLite)
require(ggplot2)
require(plyr)
require(reshape2)
require(gridExtra)
require(stringr)

#sourcing required functions
source('../Functions/function-data cleanup.R')
source('../Functions/function-descriptive statistics.R')
source('../Functions/function-scaling.R')
source('../Functions/function-extract legend.R')
source('../Functions/function-multiplot.R')
source('../Functions/function-heat map bin.R')
source('../Functions/function-defined binning.R')
source('../Functions/function-siegel tukey test.R')


#setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/LG Manuscript A')

#Input data
-------------------------------------------------------------------------------
# Choose driver
drv <- dbDriver('SQLite')

# Create connection (for SQLite, this is a file)
con <- dbConnect(drv, '../Data source files/data.db')

input <- dbGetQuery(con, "SELECT q.sampleID, q.donor, q.run, q.vessel, q.DPI,
                    q.hour, q.phase, q.dose, q.treatment,q.gen_treat, q.expt,
                    d.variable, d.value, c.Type
                    FROM Data d INNER JOIN Qualifier q
                    ON d.file = q.file
                    INNER JOIN Rank r
                    ON d.variable = r.Compound AND q.expt = r.expt
                    INNER JOIN Cmpd_type c
                    ON r.Compound = c.Compound
                    WHERE r.Confidence = 1 AND q.gen_treat = 'None'
                    AND (q.run != 8 OR q.run IS NULL)
                    AND d.variable NOT IN ('Xanthine','Glycerol')
                    AND q.state = 'ss'
                    AND q.donor != 'RP1'
                    AND q.expt != 'RP2'
                    ORDER BY q.sampleID;")

cmpd.info <- dbGetQuery(con, "SELECT c.Compound, c.Type, p.Pathway, m.main
                        FROM Cmpd_type c INNER JOIN Pathway p
                        ON c.Compound = p.Compound
                        INNER JOIN Main_pathway m
                        ON p.Pathway = m.path
                        ORDER BY c.Compound")

dbDisconnect(con)

#data cleanup
-------------------------------------------------------------------------------

#Performing data cleanup to remove zero value compounds
#also casts data into wide format
clean <- data_cleanup(input, convert.NA=TRUE)
```

```
69
70 #make sure all compounds have a value for each donor
71 data.cast <- dcast(clean, formula=sampleID+donor+run+vessel+DPI+hour+phase+dose+
72                       treatment+gen_treat+expt~variable, value.var='value')
73
74 data.cast[is.na(data.cast)] <- 0
75
76 #melting back to long format
77 data.melt <- melt(data.cast, id.vars=c('sampleID','donor','run','vessel',
78                                        'DPI','hour','phase','dose','treatment',
79                                        'gen_treat', 'expt'))
80
81 #descriptive stats
    ------------------------------------------------------------------------------
82 #Getting descriptive stats for each donor
83 stat <- ddply(data.melt, 'donor', descriptive.stat, data.type='long', variable='
       variable',
84               qualifier=NULL, scale.data=FALSE)
85
86 #Metabolite markers------------------------------------------------------------
87 #markers identified visually
88 coi <- c('3-Phenylpropionate','Leucine','Valine','Isoleucine',
89          'Methylamine','Methionine','Phenylalanine','Methanol','Succinate','
              Dimethylamine', ##Donor A
90
91          'Pimelate','Sebacate','4-Hydroxyphenylacetate','Acetate','Ethanol',
92          'Valerate','Lactate', ##Donor B
93
94          'beta-Alanine','Pyruvate', 'Glucose',##Donor C, low in Lactate too
95
96          'Tartrate','Urea','Glutamate','Pyroglutamate','Asparagine','Glycolate',
97          'Choline','Thymine','Formate', 'Isovalerate','Isobutyrate','p-Cresol',
98          'Lysine') ##RP2
99
100 coi <- factor(coi, levels=coi, ordered=TRUE)
101
102 #Seeing if same compounds picked by anova
      ---------------------------------------
103 #tukey fails becuase too few data points
104 source('../Functions/function-tukey.R')
105 source('../Functions/function-anova custom.R')
106
107 anova <- ddply(stat, 'variable', aov_custom, group='donor')
108 s.anova <- anova[anova$p.val <= 0.05 & !is.na(anova$p.val),]
109
110 tukey <- ddply(stat,'variable',tukey_custom, group='donor')
111 s.tukey <- tukey[tukey$p.val <= tukey$bon.val | is.na(tukey$p.val),]
112 #write.csv(file='Results/tukey-sig.csv', s.tukey)
113
114 #see if bonferroni correction picked anything not in coi
115 bon.tukey <- s.tukey[!s.tukey$variable %in% coi,]
116
117 #Check if bonferroni correction cut out any coi
118 cut.tukey <- coi[!coi %in% unique(s.tukey$variable)]
119 cut.tukey <- tukey[tukey$variable %in% cut.tukey,]
120
121 ##bonferroni cut out compounds found only in one donor AND had large spread
122 ##these compounds still need to be discussed
123
124 write.csv(file='./Results/profile diversity-tukey bonferroni.csv', s.tukey)
125
126 #getting compounds where one group is significantly different from every single
       group
127 marker_custom <- function(data) {
128   donor <- c(as.character(data$g1), as.character(data$g2))
129   t <- as.data.frame(table(donor))
130   t$variable <- unique(data$variable)
131   marker <- t[t$Freq == 3,]
```

```
  out <- marker[,c('variable','donor')]
  return(out)
}

marker.tukey <- ddply(s.tukey, 'variable', marker_custom)

#Getting markers that only have one donor that is different from every single other
    group
table.marker <- as.data.frame(table(as.character(marker.tukey$variable)))
uni.marker <- as.character(table.marker$Var1[table.marker$Freq == 1])

marker.tukey2 <- marker.tukey[marker.tukey$variable %in% uni.marker,]

#compare to those done by visual inspection


write.csv(file='./Results/profile diversity-marker_tukey.csv', marker.tukey)
write.csv(file='./Results/profile diversity-marker2_tukey.csv', marker.tukey2)
#Seeing if same compounds picked by t-test
    ----------------------------------------
#t-test with Bonferroni correction
t_custom <- function(data, group) {

  grps <- unique(data[,group])
  pair <- combn(grps,2)

  out <- c()
  for(i in 1:ncol(pair)) {
    grp1 <- data[data[,group] == pair[1,i],]
    grp2 <- data[data[,group] == pair[2,i],]
    grp <- rbind(grp1, grp2) #both groups being compared

    result <- t.test(grp1$value, grp2$value)
    p.val <- result$p.value
    statistic <- result$statistic
    output <- data.frame(g1=pair[1,i], g2=pair[2,i], stat=statistic, p.val=p.val)

    out <- rbind(out, output)
  }
  return(out)
}

t.result <- ddply(stat,'variable', t_custom, group='donor')

pair <- combn(unique(stat$donor),2)
bonferroni <- 0.05/ncol(pair)

sig.t <- t.result[t.result$p.val <= bonferroni & !is.na(t.result$p.val),]

#see if bonferroni correction picked anything not in coi
bon <- sig.t[!sig.t$variable %in% coi,]

#Check if bonferroni correction cut out any coi
cut <- coi[!coi %in% unique(sig.t$variable)]
cut <- t.result[t.result$variable %in% cut,]

other0s <- t.result[t.result$variable %in% c('Glucose'),]
##bonferroni cut out compounds found only in one donor AND had large spread
##these compounds still need to be discussed

write.csv(file='./Results/profile diversity-t bonferroni.csv', sig.t)

#getting compounds where one group is significantly different from ever single
    group
marker_custom <- function(data) {
  donor <- c(as.character(data$g1), as.character(data$g2))
  t <- as.data.frame(table(donor))
  t$variable <- unique(data$variable)
```

```
  print(t)
  marker <- t[t$Freq == 3,]
  out <- marker[,c('variable','donor')]
  return(out)
}

marker <- ddply(sig.t, 'variable', marker_custom)

#Getting markers that only have one donor that is different from every single other
    group
table.marker <- as.data.frame(table(as.character(marker$variable)))
uni.marker <- as.character(table.marker$Var1[table.marker$Freq == 1])

marker2 <- marker[marker$variable %in% uni.marker,]

#compare to those done by visual inspection
cut.sigt <- coi[!coi %in% unique(sig.t$variable)] #cut by bonferroni
t.cutsig <- t.result[t.result$variable %in% cut.sigt,]
cut.m1 <- coi[!coi %in% unique(marker$variable)] #cut by marker definition
t.cut1 <- sig.t[sig.t$variable %in% cut.m1,]
cut.m2 <- coi[!coi %in% unique(marker2$variable)] #cut by marker for one donor
t.cut2 <- sig.t[sig.t$variable %in% cut.m2,]

write.csv(file='./Results/profile diversity-marker.csv', marker)
write.csv(file='./Results/profile diversity-marker2.csv', marker2)
#Getting plot data and order-----------------------------------------------

p2.data <- stat[stat$variable %in% coi,]
p2.data$variable <- factor(p2.data$variable, levels=levels(coi))

pattern <- '(.*Hydroxy|henyl)(?!alanine)(.*)'
match <- '\\1-\n\\2'

new.values <- as.character(gsub(pattern, match, p2.data$variable, perl=TRUE))
new.levels <- as.character(gsub(pattern, match, levels(p2.data$variable), perl=TRUE
    ))

p2.data$wrap <- factor(new.values, levels=new.levels)

#Plotting out data values-------------------------------------------------
p2 <- ggplot(p2.data, aes(x=donor, y=value)) +
  geom_errorbar(aes(ymax=ymax, ymin=ymin)) +
  geom_point(size=1.5) +
  facet_wrap(~wrap, scales='free', ncol=5) +
  theme_bw(10) +
  scale_x_discrete(labels=c('A','B','C','    MET-2')) +
  ylab('Concentration (mM)') +
  xlab('Donor')
ggsave('./Plots/Profile diversity/markers3.png', p2,
       height=24, width=7*2.54, units='cm')

tiff(file='C:/Users/Sandi/Dropbox/Projects/Liquid Gold/Manuscript/Manuscript A/
    Journal of Proteome Research/Figure 4.tif',
         height=24/2.54, width=7, units='in', res=300,)
print(p2)
dev.off()
```

C.5 Analysis of Perturbations of Human Fecal Microbiota Propagated In Vitro

PCA analysis.R

```
#////////////////////////////////////////////////////////////////////////////
#Modified for Manuscript B
```

```
3
4  #Analysis to compare samples across projects
5
6  #loading requried packages
7  require(RSQLite)
8  require(ggplot2)
9
10 #sourcing required functions
11 setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions/')
12 source('function-pca.R')
13 source('function-plotting 2D score.R')
14 source('function-plotting 2D loading.R')
15 source('function-pca diagnostics.R')
16 source('function-data cleanup.R')
17 source('function-scaling.R')
18
19 #setting working directory
20 setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/LG Manuscript B'
       )
21
22 #-----------------------------------------------------------------------------
23 #Input data
24 # Choose driver
25 drv <- dbDriver('SQLite')
26
27 # Create connection (for SQLite, this is a file)
28 con <- dbConnect(drv, '../Data source files/data.db')
29
30 input <- dbGetQuery(con, "SELECT q.sampleID, q.donor, q.run, q.vessel, q.DPI,
31                       q.hour, q.phase, q.dose, q.treatment,q.gen_treat, q.expt,
32                       d.variable, d.value
33                       FROM Data d INNER JOIN Qualifier q
34                       ON d.file = q.file
35                       INNER JOIN Rank r
36                       ON d.variable = r.Compound AND q.expt = r.expt
37                       WHERE r.Confidence = 1 AND q.gen_treat = 'None'
38                       AND d.variable NOT IN ('Xanthine','Glycerol')
39                       AND q.state = 'ss'
40                       AND q.donor NOT IN ('C','RP1')
41                       AND q.expt != 'RP2'
42                       ORDER BY q.sampleID;")
43
44 input2 <- dbGetQuery(con, "SELECT q.sampleID, q.donor, q.run, q.vessel, q.DPI,
45                       q.hour, q.phase, q.dose, q.treatment,q.gen_treat, q.expt,
46                       d.variable, d.value
47                       FROM Data d INNER JOIN Qualifier q
48                       ON d.file = q.file
49                       INNER JOIN Rank r
50                       ON d.variable = r.Compound AND q.expt = r.expt
51                       WHERE r.Confidence = 1
52                       AND d.variable NOT IN ('Xanthine','Glycerol')
53                       AND q.state = 'ss'
54                       AND q.donor NOT IN ('C','RP1')
55                       AND q.expt != 'RP2'
56                       ORDER BY q.sampleID;")
57 dbDisconnect(con)
58
59 #setting working directory
60 setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/LG Manuscript B/
       Analysis')
61
62 #-----------------------------------------------------------------------------
63 #Performing clearnup function to remove DSS, contaminants and compounds of
64 #zero concentration
65
66 clean.data <- data_cleanup(input2, 'sampleID', 'variable','value',
67                            form.y=c('sampleID','donor','run','vessel','DPI',
68                                     'hour','phase','dose','treatment','gen_treat',
```

```
                                        'expt'),
                              form.x='variable', contaminants=NULL, export=FALSE)

#-------------------------------------------------------------------------------
#converting factors to appropriate classes
clean.data$run <- as.character(clean.data$run)
clean.data$gen_treat <- factor(clean.data$gen_treat,
                               levels=c('None','Norepinephrine','RePOOPulate',
                                        'Clindamycin+RePOOPulate','Clindamycin',
                                        'Vancomycin'))
#-------------------------------------------------------------------------------
#performing the PCA
pca.data = perform_pca(clean.data, sampleID='sampleID',
                       variable='variable', value='value',
                       form.y=c('sampleID','donor','run','vessel','DPI',
                                'hour','phase','dose','treatment','gen_treat',
                                'expt'),
                       form.x='variable',
                       scale='UV', center=TRUE, convert.NA=TRUE)
#-------------------------------------------------------------------------------
#performing diagnostics on PCA
diagnostic.results = scree_plot(pca.data)

#unpacking the diagnostic results
diagnostic.number = diagnostic.results[['number']]
diagnostic.explained = diagnostic.results[['explained']]
diagnostic.plot = diagnostic.results[['plot']]
diagnostic.plot = diagnostic.plot + ggtitle('Scree Plot (UVN)')

#printing out diagnostic results
print(diagnostic.number)
print(diagnostic.explained)
print(diagnostic.plot)
#-------------------------------------------------------------------------------
#plotting PCA-processed data
#Score plot 2D
#   Note: Arguments for score plot function are
#           (pca_data, qualifier_data, PCs=1:4,colour=NULL, shape=NULL, size=NULL,
#           label=NULL, title=NULL, legend.ori='horizontal', black_white=FALSE)

score_plot = score_plot_2d(pca.data, PCs=1:4,
                           colour='gen_treat', shape='donor',
                           legend.ori='horizontal', outline=FALSE,
                           black_white=FALSE)

p.score <- score_plot[['p']]
collected.data <- score_plot[['collected.data']]

ellipse_custom <- function(d) {

  #Calculating ellipse for 95% confidence
  ellipseSD = cor(d$x.value, d$y.value)
  ellipseDf = as.data.frame(ellipse(ellipseSD, scale=c(sd(d$x.value),
                                                       sd(d$y.value)),
                                    centre=colMeans(d[,c('x.value','y.value')]),
                                    t=2))
  return(ellipseDf)
}

e.path <- ddply(collected.data, 'view', ellipse_custom)


#Drawing an ellipse for 95% confidence
p.score = p.score +
  #  geom_path(data=e.path, aes(x=x, y=y), linetype=2, alpha=0.5) +
  #  scale_colour_manual(values=c('black','darkorange1','cornflowerblue',
  #                               'black','chartreuse','blue', 'grey30'),
  #                      guide_legend(title='Run'),
```

```
  #                              labels=c('NA','1','2','3','4','5','6')) +
  scale_shape_manual(name='Donor', labels=c('A','B','MET-2'), values=c(16,7,5)) +
  scale_colour_manual(values=c('cyan4','darkorchid4','chartreuse',
                               'darkorange1','deeppink1','blue', 'grey30'),
                      labels=c('None','Norepinephrine','MET-1',
                               'Clindamycin+MET-1','Clindamycin',
                               'Vancomycin'),
                      guide_legend(title='Treatment',
                                   )) +
  guides(col=guide_legend(byrow=TRUE,nrow=2))

ggsave('../Plots/PCA plots/treatment-score.png', p.score, height=15, width=25,
    units='cm')

#     ------------------------------------------------------------------------------------------------------------------------>


#Loading plot 2D
#   Note: Arguments for loading plot function are (pca_data, PCs=1:4, label=NULL,
    title=NULL,
#                                                  view_mode='compressed')

#If you want all compound labels shown, enter all_paths in label argument
all_paths = rownames(pca.data$rotation)

#If you want all and only compound within the subset of observations processed in
    the PCA function, can create a temperary group called 'a' to enter in for the
    label argument
#a = pathway_list$Carb[pathway_list$Organic %in% rownames(pca_processed_data$
    rotation)]

loading_plot = loading_plot_2d(pca.data, PCs=1:4, label=all_paths, outline=TRUE,
                               title=NULL, view_mode = 'compressed')

p.load <- loading_plot[['p']]
load.data <- loading_plot[['collected.data']]

ggsave('../Plots/PCA plots/treatment-loading.png', p.load,
       height=14.5*1.5, width=25*2, units='cm')
```

Treatment analysis.R

```
#////////////////////////////////////////////////////////////////////////////////
    /////
#Modified for Manuscript B
#Antibiotic treatment analysis
#Heat map

#loading required packages
require(ggplot2)
require(RODBC)
require(RSQLite)
require(gridExtra)
require(scales)

#sourcing required functions
setwd('c:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions/')
source('function-descriptive statistics.R')
source('function-data cleanup.R')
source('function-extract legend.R')
source('function-bin function.R')


#setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/LG Manuscript B/
    Analysis')
source('../Functions/function-defined binning.R')
#-------------------------------------------------------------------------------
```

```
#Input data
con <-  odbcConnect('MySQLDSN')

input <- sqlQuery(con, "SELECT q.sampleID, q.donor, q.run, q.vessel, q.DPI,
                       q.hour, q.phase, q.dose, q.treatment,q.gentreat, q.expt,
                       d.variable, d.value
                       FROM liquidgold.data d INNER JOIN liquidgold.qualifier q
                       ON d.file = q.file
                       INNER JOIN liquidgold.rank r
                       ON d.variable = r.Compound AND q.expt = r.expt
                       WHERE r.Confidence = 1
                       AND (q.run != 8 OR q.run IS NULL)
                       AND d.variable NOT IN ('Xanthine','Glycerol')
                       AND q.state = 'ss'
                       AND q.donor NOT IN ('C','RP1')
                       AND q.expt != 'RP2'
                       ORDER BY q.sampleID;")

odbcClose(con)

#-------------------------------------------------------------------------
#Performing data cleanup to remove zero value compounds
#also casts data into wide format
clean <- data_cleanup(input, form.y=c('sampleID','donor','run','vessel',
                                      'DPI','hour','phase','dose','treatment',
                                      'gentreat', 'expt'),
                         form.x='variable', convert.NA=TRUE)

#make sure all compounds have a value for each donor
data.cast <- dcast(clean, formula=sampleID+donor+run+vessel+DPI+hour+phase+dose+
                        treatment+gentreat+expt~variable, value.var='value')

#Assigning 0 to where compound was not profiled
data.cast[,12:ncol(data.cast)][is.na(data.cast[,12:ncol(data.cast)])] <- 0

#melting back to long format
data.melt <- melt(data.cast, id.vars=c('sampleID','donor','run','vessel',
                                       'DPI','hour','phase','dose','treatment',
                                       'gentreat', 'expt'))
#-------------------------------------------------------------------------
#Getting descriptive stats for  compounds within gentreat for each donor
stat <- ddply(data.melt, c('donor','gentreat'), descriptive.stat, data.type='long',
              variable='variable')

#t test
#-------------------------------------------------------------------------
#testing if the change from non-treated to treated is significant usint t-test

t_custom <- function(data, gentreat) {

  none <- data[data$gentreat == 'None',]
  out <- none
  out$p.val <- NA

  for(i in gentreat){

    treat <- data[data$gentreat == i,]

    #when treated samples have more than 1 value, can use comparison of 2 means
    if(length(unique(treat$sampleID)) > 1) {
      p.value=t.test(none$value, treat$value)$p.value
      output <- treat
      output$p.val <- p.value

      out <- rbind(out, output)
    }

    #when treated samples have only 1 value, do 1-sample t test, where treated
```

```
        value
92    #is the true mean
93    else if(length(unique(treat$sampleID)) == 1) {
94      x.bar <- mean(none$value)
95      y.value <- treat$value
96      x.sd <- sd(none$value)
97      n <- length(unique(none$sampleID))
98
99      t.stat <-  (x.bar - y.value)/(x.sd/sqrt(n))
100     p.value <- 2*pt(-abs(t.stat), n-1)
101
102     output <- treat
103     output$p.val <- p.value
104
105     out <- rbind(out, output)
106   }
107
108   #when the gentreat is not in that donor
109   else if(length(unique(treat$sampleID)) < 1) {
110     out <- out
111   }
112 }
113 return(out)
114 }
115
116 t.result <- ddply(stat, c('donor','variable'),
117                   t_custom, c('Clindamycin','RePOOPulate',
118                               'Clindamycin+RePOOPulate','Norepinephrine','
                                    Vancomycin'))
119 #write.csv(file='check tresult.csv', t.result)
120
121 #Mann Whitney test
    ------------------------------------------------------------------------
122 #testing if the change from non-treated to treated is significant using mann
        whitney test
123
124 mw_custom <- function(data, gentreat) {
125
126   none <- data[data$gentreat == 'None',]
127   out <- none
128   out$p.val <- NA
129
130   for(i in gentreat){
131
132     treat <- data[data$gentreat == i,]
133
134     #when treated samples have more than 1 value, can use comparison of 2 means
135     if(length(unique(treat$sampleID)) > 1) {
136       p.value=wilcox.test(none$value, treat$value)$p.value
137       output <- treat
138       output$p.val <- p.value
139
140       out <- rbind(out, output)
141     }
142
143     #when treated samples have only 1 value, do 1-sample t test, where treated
          value
144     #is the true mean
145     else if(length(unique(treat$sampleID)) == 1) {
146       x.bar <- mean(none$value)
147       y.value <- treat$value
148       x.sd <- sd(none$value)
149       n <- length(unique(none$sampleID))
150
151       t.stat <-  (x.bar - y.value)/(x.sd/sqrt(n))
152       p.value <- 2*pt(-abs(t.stat), n-1)
153
154       output <- treat
```

```
155         output$p.val <- p.value
156
157         out <- rbind(out, output)
158       }
159
160       #when the gentreat is not in that donor
161       else if(length(unique(treat$sampleID)) < 1) {
162         out <- out
163       }
164     }
165     return(out)
166 }
167
168 mw.result <- ddply(stat, c('donor','variable'),
169                    mw_custom, c('Clindamycin','RePOOPulate',
170                                 'Clindamycin+RePOOPulate','Norepinephrine','
                                        Vancomycin'))
171
172 #getting difference between none and treatment means
      ----------------------------------
173
174 diff_custom <- function(data, gentreat) {
175   none <- data[data$gentreat == 'None',]
176   out <- none
177   out$uni.treat <- NA
178   out$diff <- NA
179   out$norm.diff <- NA
180   out$per.diff <- NA
181
182   for(i in gentreat) {
183     treat <- data[data$gentreat == i,]
184     if(nrow(treat) != 0) {
185       diff <- treat
186       diff$uni.treat <- paste(unique(treat$gentreat),unique(treat$donor), sep='_')
187       diff$diff <- unique(treat$avg)-unique(none$avg)
188       diff$norm.diff <- (unique(treat$avg)-unique(none$avg))/(unique(none$avg)+
              unique(treat$avg))/2*100
189       diff$per.diff <- (unique(treat$avg)-unique(none$avg))/unique(none$avg)*100
190
191       out <- rbind(out, diff)
192     }
193     else if(nrow(treat) == 0) out <- out
194   }
195   return(out)
196 }
197
198 diff <- ddply(mw.result, c('donor','variable'),
199               diff_custom, c('Clindamycin','RePOOPulate',
200                              'Clindamycin+RePOOPulate','Norepinephrine','Vancomycin
                                    '))
201
202 #removing none rows-------------------------------------------------------------
203
204 diff <- diff[diff$gentreat != 'None',]
205
206 #giving artificial diff values where non significant differences
207 #are assigned percent difference of zero
208 diff$sig.perdiff <- diff$per.diff
209 diff$sig.perdiff[diff$p.val > 0.05] <- 0
210
211 #artificially turning compounds with hi sig.perdiff into NA to make
212 #it a different colour when plotting
213 diff$sig.perdiff[diff$sig.perdiff > 100 & !is.na(diff$sig.perdiff)] <- NA
214
215 #zero compounds
      ------------------------------------------------------------------------
216
217 #Getting compounds with zero concentration in none or treated, but not both
```

```
zero.cmpd <- unique(as.character(diff$variable[!is.na(diff$per.diff) &
                                                 (diff$per.diff == -100 | diff$per.
                                                     diff == 'Inf')]))


zero.data <- diff[diff$variable %in% zero.cmpd,]

bin.list <- defined_binning()

#setting working directory back
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/LG Manuscript B/
    Analysis')

zero.order <- list()
for(i in 1:length(bin.list)) {
  temp <- zero.data[zero.data$variable %in% bin.list[[i]],]
  temp <- unique(temp[,c('uni.treat','variable','diff')])
  temp$variable <- factor(temp$variable, levels=rev(levels(bin.list[[i]])), ordered
      =TRUE)
  temp$uni.treat <- factor(temp$uni.treat,
                           levels=c('Norepinephrine_A','RePOOPulate_A',
                                    'Clindamycin+RePOOPulate_A',
                                    'Clindamycin_A','Clindamycin_B','Vancomycin_RP2
                                        '),
                           ordered=TRUE)
  temp <- temp[order(temp$variable),]
  temp <- temp[order(temp$uni.treat),]

  zero.order[[i]] <- temp
}

zero.export <- do.call(rbind.data.frame, zero.order)
write.csv(file='../Results/zero.data.csv', zero.export)

#hi compounds
-------------------------------------------------------------------------------
#getting compounds that have sig.perdiff > 100
hi.cmpd <- unique(as.character(diff$variable[diff$per.diff > 100 &
                                               !is.na(diff$per.diff)]))

hi.data <- diff[diff$variable %in% hi.cmpd,]
hi.data <- hi.data[!hi.data$variable %in% zero.cmpd,]

hi.order <- list()
for(i in 1:length(bin.list)) {
  temp <- hi.data[hi.data$variable %in% bin.list[[i]],]
  temp <- unique(temp[,c('uni.treat','variable','per.diff')])
  temp$variable <- factor(temp$variable, levels=rev(levels(bin.list[[i]])), ordered
      =TRUE)
  temp$uni.treat <- factor(temp$uni.treat,
                           levels=c('Norepinephrine_A','RePOOPulate_A',
                                    'Clindamycin+RePOOPulate_A',
                                    'Clindamycin_A','Clindamycin_B','Vancomycin_RP2
                                        '),
                           ordered=TRUE)
  temp <- temp[order(temp$variable),]
  temp <- temp[order(temp$uni.treat),]

  hi.order[[i]] <- temp
}

for(i in 1:length(hi.order)){
  hi.write <- hi.order[[i]]
  write.csv(file=paste('../Results/hi.data',i,'.csv', sep=''), hi.write)
}
#density plots
-------------------------------------------------------------------------------
#looking at density plot of mean concentrations
```

```
plot(density(stat.diff$avg))
plot(density(stat.zero$avg))
plot(density(stat.hi$avg))
plot(density(diff.data$sig.perdiff[!is.na(diff.data$sig.perdiff)]))

#Assigning diff data to bins as defined by none-treated samples
    ----------------------------

bin.none <- bin.list
bin.treat <- bin_function(diff,bin=c(20,10,1,0.1,0))
bin.treat <- do.call(c,bin.treat)

none.cmpd <- c()
treat.cmpd <- c()
for(i in 1:length(bin.none)) {
  temp <- as.character(bin.none[[i]])
  temp2 <- as.character(bin.treat[[i]])
  none.cmpd <- c(none.cmpd, temp)
  treat.cmpd <- c(treat.cmpd, temp2)
}

#check if any compouds in treated samples not in non-treated samles
# if no can just use none sample binning
if(any(!treat.cmpd %in% none.cmpd)==FALSE) bin.list <- bin.none

#if yes, then need to augment none sample binning to include those compounds
#in the right orer
####write when needed####

#Setting treatment order-----------------------------------------------------------

diff$gentreat <- factor(diff$gentreat,
                        levels=c('Norepinephrine','RePOOPulate',
                                 'Clindamycin+RePOOPulate',
                                 'Clindamycin','Vancomycin'))

diff$uni.treat <- factor(diff$uni.treat,
                        levels=c('Norepinephrine_A','RePOOPulate_A',
                                 'Clindamycin+RePOOPulate_A',
                                 'Clindamycin_A','Clindamycin_B','Vancomycin_RP2')
                                     )

diff$pad <- sprintf('%022s',diff$variable)
#diff$variable <- factor(diff$variable, levels=none.cmpd, ordered=TRUE)

pad.list <- list()
for(i in 1:length(bin.none)) {
  pad.temp <- sprintf('%022s', as.character(bin.list[[i]]))
  pad.list[[i]] <- pad.temp
}

pad.order <- c()
for(i in 1:length(pad.list)) {
  temp <- as.character(pad.list[[i]])
  pad.order <- c(pad.order, temp)
}

#hi compounds for table
    2-----------------------------------------------------------------
#Compounds with percent change > 100%
hi.cmpd <- unique(diff$variable[diff$per.diff > 100 & !is.na(diff$per.diff)])

#diff data for those compounds
out <- unique(diff[diff$variable %in% hi.cmpd, c('variable','uni.treat','per.diff',
    'p.val')])

#converting per.diff to zero if insignificant
out$per.diff[out$p.val > 0.05] <- 0
```

```
#transforming into casted form
out2 <- dcast(out, variable~uni.treat, value.var='per.diff')

#re-ordering columns to particular order
out3 <- out2[,c('variable','Norepinephrine_A','RePOOPulate_A','Clindamycin+
    RePOOPulate_A',
                'Clindamycin_A','Clindamycin_B','Vancomycin_RP2')]

#rearranging compounds to bin order
out3$variable <- factor(out3$variable, levels=none.cmpd, ordered=TRUE)
out3 <- out3[order(out3$variable),]

#Adding compound classes
type.info <- unique(cmpd.info[cmpd.info$Compound %in% hi.cmpd,1:2])
type.info$Compound <- factor(type.info$Compound, levels=none.cmpd, ordered=TRUE)
type.info <- type.info[order(type.info$Compound),]

out3$class <- type.info$Type

path.info <- cmpd.info[cmpd.info$Compound %in% hi.cmpd,]

#export as csv
#write.csv(file='../Results/hi percent difference.csv',out3)
write.csv(file='../Results/path info.csv', path.info)

#exporting hi and zero data
-------------------------------------------------------------------------------
#exporting hi data and zero data to csv file
zero.data2 <- unique(zero.data[,c('variable','uni.treat','med','diff')])


#ordering zero.data2 so variables are in order of median values
order.zero2 <- zero.data2[order(zero.data2$med , decreasing=TRUE),]
order.zero2$variable <- factor(order.zero2$variable,
                               levels=rev(unique(order.zero2$variable)),
                               ordered=TRUE)
zero.ordered2 <- zero.data2
zero.ordered2$variable <- factor(zero.ordered2$variable,
                                 levels=levels(order.zero2$variable))
zero.ordered2$uni.treat <- factor(zero.ordered2$uni.treat,
                                  levels=c('Norepinephrine_A','RePOOPulate_A',
                                           'Clindamycin+RePOOPulate_A',
                                           'Clindamycin_A','Clindamycin_B','
                                               Vancomycin_RP2'))

zero.out <- dcast(zero.data2, formula=variable~uni.treat, value.var='diff')

#write.csv(file='../Plots/zero data.csv', zero.out)


hi.data2 <- unique(hi.data[,c('variable','uni.treat','med','sig.perdiff')])

#ordering hi.data2 so variables are in order of median values
order.hi2 <- hi.data2[order(hi.data2$med , decreasing=TRUE),]
order.hi2$variable <- factor(order.hi2$variable, levels=rev(unique(order.hi2$
    variable)),
                             ordered=TRUE)
hi.ordered2 <- hi.data2
hi.ordered2$variable <- factor(hi.ordered2$variable, levels=levels(order.hi2$
    variable))
hi.ordered2$uni.treat <- factor(hi.ordered2$uni.treat,
                                levels=c('Norepinephrine_A','RePOOPulate_A',
                                         'Clindamycin+RePOOPulate_A',
                                         'Clindamycin_A','Clindamycin_B','
                                             Vancomycin_RP2'))

hi.out <- dcast(hi.ordered2, formula=variable~uni.treat, value.var='sig.perdiff')

```

```
#write.csv(file='check hi data.csv', hi.data)
#write.csv(file='check zero data.csv', zero.data)
#write.csv(file='../Plots/hi data.csv', hi.out)

#binning hi data by sig.perdiff
#-------------------------------------------------------------------------------
#binning hi.data by sig.perdiff
hi.data$bin <- NA
hi.data[hi.data$sig.perdiff >= 1000  & !is.na(hi.data$sig.perdiff), 'bin'] <- 'bin1'
hi.data[hi.data$sig.perdiff >= 500 & hi.data$sig.perdiff < 1000 &
          !is.na(hi.data$sig.perdiff), 'bin'] <- 'bin2'
hi.data[hi.data$sig.perdiff >= 100 & hi.data$sig.perdiff < 500 &
          !is.na(hi.data$sig.perdiff), 'bin'] <- 'bin3'

#converting compounds to character class and also gettinng compounds in each bin
bin.cmpd <- list()
for(i in c('bin1','bin2','bin3')) {
  temp <- hi.data[hi.data$bin == i & !is.na(hi.data$bin),]
  bin.cmpd[[i]] <- as.character(unique(temp$variable))
}

#Getting only compounds found in one bin and not any other bins before it
common.cmpd <- c(bin.cmpd[[1]])
hi.list <- list(bin.cmpd[[1]])

for(i in 2:length(bin.cmpd)) {
  hi.list[[i]] <- bin.cmpd[[i]][!bin.cmpd[[i]] %in% common.cmpd]
  common.cmpd <- c(common.cmpd, hi.list[[i]])
}
#ordering hi.data so variables are in order of median values
order.hi <- hi.data[order(hi.data$med , decreasing=TRUE),]
order.hi$variable <- factor(order.hi$variable, levels=rev(unique(order.hi$variable)),
                            ordered=TRUE)
hi.ordered <- hi.data
hi.ordered$variable <- factor(hi.ordered$variable, levels=levels(order.hi$variable))

hi.ordered$gentreat <- factor(hi.ordered$gentreat,
                              levels=levels(ordered.data$gentreat))
hi.ordered$uni.treat <- factor(hi.ordered$uni.treat,
                              levels=levels(ordered.data$uni.treat))

#plotting----------------------------------------------------------------------

plot.list <- list()
legend.list <- list()
fig.list <- list()
fig.leg <- list()
leg.lim <- list()

leg.width <- c(10,10,10,10,10)
leg.pos <- list(c(0.5, 0.5), c(0.5, 0.5), c(0.5, 0.5), c(0.5, 0.5), c(0.6, 0.45))
fig.width <- c(8.6, 8.5, 8.6, 9.2, 9.4)
fig.height <- c(2.4,2.8,4.2,13.5,11.2)

margin <- list(c(0.1,0.8,0.9,0.8),
               c(0.1,0.8,1,1),
               c(0.1,0.8,0.9,0.5),
               c(0,0.2,0,0),
               c(0,0.2,0,0))

#Panels 1:3--------------------------------------------------------------------
for(i in 1:3) {
  plot.data <- unique(diff[diff$variable %in% bin.list[[i]],
                           c('variable','pad','gentreat','uni.treat','p.val',
                             'avg','med','diff','per.diff','sig.perdiff')])
```

```
468
469   #Setting compound order
470   plot.data$variable <- factor(as.character(plot.data$variable),
471                                levels=rev(none.cmpd[none.cmpd %in% bin.list[[i]]]),
472                                ordered=TRUE)
473   plot.data$pad <- factor(as.character(plot.data$pad),
474                           levels=rev(pad.order[pad.order %in% pad.list[[i]]]),
                                ordered=TRUE)
475
476
477   plot.data$label <- '*'
478
479   #write.csv(file=paste('../Results/check plot data',i,'.csv',sep=''), temp)
480   #getting legend limits
481   max.lim <- max(plot.data$sig.perdiff[!is.na(plot.data$sig.perdiff)])
482   min.lim <- min(plot.data$sig.perdiff[!is.na(plot.data$sig.perdiff)])
483   larger.lim <- max(abs(max.lim), abs(min.lim))
484   leg.lim[[i]] <- c(-larger.lim, larger.lim)
485   print(max.lim)
486   print(min.lim)
487
488   plot.list[[i]] <- ggplot(plot.data, aes(x=uni.treat, y=pad, label=label))
489
490   plot.list[[i]] <- plot.list[[i]] + geom_tile(fill=NA, colour='white')
491
492   if(any(is.na(plot.data$sig.perdiff))==FALSE) {
493     plot.list[[i]] <- plot.list[[i]] +
494       geom_tile(aes(fill=sig.perdiff), colour='white')
495     print('here1')
496   }
497
498   else if(any(is.na(plot.data$sig.perdiff))==TRUE) {
499     plot.list[[i]] <- plot.list[[i]] +
500
501       #fill tile by sig.perdiff
502       geom_tile(data=plot.data[!is.na(plot.data$sig.perdiff),],
503                 aes(fill=sig.perdiff),colour='white')
504
505     if(nrow(plot.data[plot.data$p.val <=0.05 &
506                         plot.data$per.diff == -100 &
507                         !is.na(plot.data$per.diff),]) > 0) {
508
509       plot.list[[i]] <- plot.list[[i]] +
510
511         #compounds with percent diff of -100% coloured with red with *
512         geom_tile(data=plot.data[plot.data$p.val <=0.05 &
513                                    plot.data$per.diff == -100 &
514                                    !is.na(plot.data$per.diff),],
515                   colour='white', fill='red') +
516         geom_text(data=plot.data[plot.data$p.val <=0.05 &
517                                    plot.data$per.diff == -100 &
518                                    !is.na(plot.data$per.diff),],
519                   colour='white', size=6, hjust=0.5, vjust=0.8)
520     }
521
522     if(nrow(plot.data[(plot.data$p.val <= 0.05 &
523                          plot.data$per.diff > 100 & !is.nan(plot.data$sig.perdiff))
                                |
524                          plot.data$per.diff == 'Inf',]) > 0) {
525
526       plot.list[[i]] <- plot.list[[i]] +
527
528         #compounds percent diff > 100% or Inf (0 in none, 1 in treated) coloured
              blue with *
529         geom_tile(data=plot.data[(plot.data$p.val <= 0.05 &
530                                     plot.data$per.diff > 100 & !is.nan(plot.data$
                                          sig.perdiff)) |
531                                   plot.data$per.diff == 'Inf',],
```

```
532                     colour='white',fill='blue') +
533           geom_text(data=plot.data[(plot.data$p.val <= 0.05 &
534                                       plot.data$per.diff > 100 & !is.nan(plot.data$
                                           sig.perdiff)) |
535                                      plot.data$per.diff == 'Inf',],
536                     colour='white', size=6, hjust=0.5, vjust=0.8)
537       }
538
539       print('here2')
540
541     }
542 #-------------------------------------------------------------------
543     plot.list[[i]] <- plot.list[[i]] +
544       scale_x_discrete(labels=c('Norepinephrine','MET-1','Clindamycin+\nMET-1',
545                                 'Clindamycin (A)','Clindamycin (B)','Vancomycin')) +
546       scale_fill_gradientn(colours=c('red','white','blue'),
547                            values=rescale(c(-larger.lim,0,larger.lim)),
548                            name='Percent Difference\nin Concentration\t',
549                            limits=leg.lim[[i]]) +
550       theme_bw(12) +
551       theme(panel.background=element_rect('grey60'),
552             panel.grid.major=element_line(colour = 'grey40'),
553             axis.text.x=element_blank(),
554             axis.ticks.x=element_blank(),
555             axis.title.x=element_blank(),
556             axis.title.y=element_blank(),
557             legend.key.width=unit(1.1, 'cm'),
558             legend.key.height=unit(0.35,'cm'),
559             legend.position=c(0.55,0.4),legend.direction='horizontal',
560             legend.text=element_text(size=7.5),
561             legend.title=element_text(size=7.5),
562             plot.margin=unit(margin[[i]], 'cm'))
563
564     legend.list[[i]] <- g_legend(plot.list[[i]])
565
566     #Export as panels-------------------------------------------------
567     if(FALSE){
568       #export as EPS; panels with legend
569       setEPS()
570       postscript(file=paste('../Results/test plot',i,'.eps', sep=''),
571                  onefile=TRUE, paper='special', horizontal=FALSE,
572                  height=fig.height[i]+1, width=fig.width[i], pointsize=10)
573       grid.arrange(legend.list[[i]], plot.list[[i]] + theme(legend.position='none'),
574                    heights=c(0.1, 0.9))
575       dev.off()
576
577       #export as EPS; panels separate from legend
578       setEPS()
579       postscript(file=paste('../Results/test plot',i,'.eps', sep=''),
580                  onefile=FALSE, paper='special', horizontal=FALSE,
581                  height=fig.height[i], width=fig.width[i], pointsize=10)
582       fig.list[[i]] <- grid.arrange(plot.list[[i]] + theme(legend.position='none'))
583       dev.off()
584
585       #export as PNG - panels
586       png(filename=paste('../Plots/treatment effects/per diff_bin',i,'.png', sep=''),
587           type='cairo', height=fig.height[i], width=fig.width[i], units='cm', res
                =240, pointsize=10)
588       fig.list[[i]] <- grid.arrange(plot.list[[i]] + theme(legend.position='none'))
589       dev.off()
590
591       png(filename=paste('../Plots/treatment effects/per diff_leg',i,'.png', sep=''),
592           height=1, width=leg.width[i], units='cm', res=240, pointsize=10)
593       fig.leg[[i]] <- grid.arrange(legend.list[[i]])
594       dev.off()
595     }
596 }
597
```

```
#panels 4 and 5-------------------------------------------------------------------
  for(i in 4:length(bin.list)) {
    plot.data <- unique(diff[diff$variable %in% bin.list[[i]],
                             c('variable','pad','gentreat','uni.treat','p.val',
                               'avg','med','diff','per.diff','sig.perdiff')])

    #Setting compound order
    plot.data$variable <- factor(as.character(plot.data$variable),
                                 levels=rev(none.cmpd[none.cmpd %in% bin.list[[i
                                     ]]]),
                                 ordered=TRUE)
    plot.data$pad <- factor(as.character(plot.data$pad),
                            levels=rev(pad.order[pad.order %in% pad.list[[i]]]),
                                ordered=TRUE)


    plot.data$label <- '*'

    #write.csv(file=paste('../Results/check plot data',i,'.csv',sep=''), temp)
    #getting legend limits
    max.lim <- max(plot.data$sig.perdiff[!is.na(plot.data$sig.perdiff)])
    min.lim <- min(plot.data$sig.perdiff[!is.na(plot.data$sig.perdiff)])
    larger.lim <- max(abs(max.lim), abs(min.lim))
    leg.lim[[i]] <- c(-larger.lim, larger.lim)
    print(max.lim)
    print(min.lim)

    plot.list[[i]] <- ggplot(plot.data, aes(x=uni.treat, y=pad, label=label))

    plot.list[[i]] <- plot.list[[i]] + geom_tile(fill=NA, colour='white')

    if(any(is.na(plot.data$sig.perdiff))==FALSE) {
      plot.list[[i]] <- plot.list[[i]] +
        geom_tile(aes(fill=sig.perdiff), colour='white')
      print('here1')
    }

    else if(any(is.na(plot.data$sig.perdiff))==TRUE) {
      plot.list[[i]] <- plot.list[[i]] +

        #fill tile by sig.perdiff
        geom_tile(data=plot.data[!is.na(plot.data$sig.perdiff),],
                  aes(fill=sig.perdiff),colour='white')

      if(nrow(plot.data[plot.data$p.val <=0.05 &
                          plot.data$per.diff == -100 &
                          !is.na(plot.data$per.diff),]) > 0) {

        plot.list[[i]] <- plot.list[[i]] +

          #compounds with percent diff of -100% coloured with red with *
          geom_tile(data=plot.data[plot.data$p.val <=0.05 &
                                     plot.data$per.diff == -100 &
                                     !is.na(plot.data$per.diff),],
                    colour='white', fill='red') +
          geom_text(data=plot.data[plot.data$p.val <=0.05 &
                                     plot.data$per.diff == -100 &
                                     !is.na(plot.data$per.diff),],
                    colour='white', size=6, hjust=0.5, vjust=0.8)
      }

      if(nrow(plot.data[(plot.data$p.val <= 0.05 &
                           plot.data$per.diff > 100 & !is.nan(plot.data$sig.perdiff
                               )) |
                           plot.data$per.diff == 'Inf',]) > 0) {

        plot.list[[i]] <- plot.list[[i]] +

```

```
663           #compounds percent diff > 100% or Inf (0 in none, 1 in treated) coloured
                  blue with *
664           geom_tile(data=plot.data[(plot.data$p.val <= 0.05 &
665                                       plot.data$per.diff > 100 & !is.nan(plot.data$
                                            sig.perdiff)) |
666                                      plot.data$per.diff == 'Inf',],
667                     colour='white',fill='blue') +
668           geom_text(data=plot.data[(plot.data$p.val <= 0.05 &
669                                       plot.data$per.diff > 100 & !is.nan(plot.data$
                                            sig.perdiff)) |
670                                      plot.data$per.diff == 'Inf',],
671                     colour='white', size=6, hjust=0.5, vjust=0.8)
672       }
673
674       print('here2')
675
676     }
677 #
      ------------------------------------------------------------------------------


678     plot.list[[i]] <- plot.list[[i]] +
679       scale_x_discrete(labels=c('Norepinephrine','MET-1','Clindamycin+\nMET-1',
680                                 'Clindamycin (A)','Clindamycin (B)','Vancomycin'))
                                    +
681       scale_fill_gradientn(colours=c('red','white','blue'),
682                            values=rescale(c(-larger.lim,0,larger.lim)),
683                            name='Percent Difference\nin Concentration\t',
684                            limits=leg.lim[[i]]) +
685       theme_bw(12) +
686       xlab('Treatment') +
687       theme(panel.background=element_rect('grey60'),
688             panel.grid.major=element_line(colour = 'grey40'),
689             axis.text.x=element_text(hjust=0, vjust=1, angle=-45),
690             axis.title.y=element_blank(),
691             plot.margin=unit(c(0.5,0.8,0,0), 'cm'),
692             legend.key.width=unit(1.1, 'cm'),
693             legend.key.height=unit(0.35,'cm'),
694             legend.position=c(0.55,0.4),legend.direction='horizontal',
695             legend.text=element_text(size=7.5),
696             legend.title=element_text(size=7.5),
697             plot.margin=unit(margin[[i]], 'cm'))
698
699     legend.list[[i]] <- g_legend(plot.list[[i]])
700
701     #Export as panels-------------------------------------------------
702     if(FALSE){
703       #export as EPS; panels with legend
704       setEPS()
705       postscript(file=paste('../Results/test plot',i,'.eps', sep=''),
706                  onefile=TRUE, paper='special', horizontal=FALSE,
707                  height=fig.height[i]+1, width=fig.width[i], pointsize=10)
708       grid.arrange(legend.list[[i]], plot.list[[i]] + theme(legend.position='none')
              ,
709                    heights=c(0.1, 0.9))
710       dev.off()
711
712       #export as EPS; panels separate from legend
713       setEPS()
714       postscript(file=paste('../Results/test plot',i,'.eps', sep=''),
715                  onefile=FALSE, paper='special', horizontal=FALSE,
716                  height=fig.height[i], width=fig.width[i], pointsize=10)
717       fig.list[[i]] <- grid.arrange(plot.list[[i]] + theme(legend.position='none'))
718       dev.off()
719
720       #export as PNG - panels
721       png(filename=paste('../Plots/treatment effects/per diff_bin',i,'.png', sep=''
              ),
722           type='cairo', height=fig.height[i], width=fig.width[i], units='cm', res
```

```
                =240, pointsize=10)
723         fig.list[[i]] <- grid.arrange(plot.list[[i]] + theme(legend.position='none'))
724         dev.off()
725
726         png(filename=paste('../Plots/treatment effects/per diff_leg',i,'.png', sep=''
                ),
727             height=1, width=leg.width[i], units='cm', res=240, pointsize=10)
728         fig.leg[[i]] <- grid.arrange(legend.list[[i]])
729         dev.off()
730       }
731 }
732
733
734 #Setting plot dimensions
    ------------------------------------------------------------------
735 widths <- list(c(10,6)/2.54+1,          #1
736                c(10,6)/2.54+1,          #2
737                c(10,6)/2.54+1,          #3
738                c(10,6)/2.54+1,          #4
739                c(10,10.4)/2.54+1)       #5
740
741 heights <- list(c(1,2.4)/2.54,            #1
742                 c(1,3)/2.54,             #2
743                 c(1,4.8)/2.54,           #3
744                 c(1,15)/2.54,           #4
745                 c(1,11.3)/2.54)          #5
746
747
748
749 #PNG-------------------------------------------------
750 png(filename='../Plots/treatment effects/Percent Difference4.png',
751     width=10.5/2.54*2, height=sum(c(heights[[4]],heights[[3]])),
752     units='in',res=240,pointsize=10)
753 grid.arrange(arrangeGrob(legend.list[[1]],
754                          plot.list[[1]] + theme(legend.position='none'),
755                          legend.list[[2]],
756                          plot.list[[2]] + theme(legend.position='none'),
757                          legend.list[[5]],
758                          plot.list[[5]] + theme(legend.position='none'),
759                          ncol=1, heights=c(heights[[1]],heights[[2]],heights[[5]]),
760                          widths=c(widths[[1]],widths[[2]],widths[[5]])),
761              arrangeGrob(legend.list[[3]],
762                          plot.list[[3]] + theme(legend.position='none'),
763                          legend.list[[4]],
764                          plot.list[[4]] + theme(legend.position='none'),
765                          ncol=1, heights=c(heights[[3]],heights[[4]]),
766                          widths=c(widths[[3]],widths[[4]])),
767              ncol=2)
768 grid.text('A', x=unit(0.5,'cm'), y=unit(21.1, 'cm'),gp=gpar(cex=1.5))
769 grid.text('C', x=unit(0.5,'cm'), y=unit(17.4, 'cm'),gp=gpar(cex=1.5))
770 grid.text('E', x=unit(0.5,'cm'), y=unit(12.95, 'cm'),gp=gpar(cex=1.5))
771 grid.text('B', x=unit(11,'cm'), y=unit(21.2, 'cm'),gp=gpar(cex=1.5))
772 grid.text('D', x=unit(11,'cm'), y=unit(15.4, 'cm'),gp=gpar(cex=1.5))
773
774 dev.off()
775
776 #EPS-------------------------------------------------
777
778 setEPS()
779 postscript(file='../Results/test plot.eps',
780            onefile=FALSE, paper='special', horizontal=FALSE,
781            width=10.5/2.54*2, height=sum(c(heights[[4]],heights[[3]])),
782            pointsize=10)
783 grid.arrange(arrangeGrob(legend.list[[1]],
784                          plot.list[[1]] + theme(legend.position='none'),
785                          legend.list[[2]],
786                          plot.list[[2]] + theme(legend.position='none'),
787                          legend.list[[5]],
```

```
788                           plot.list[[5]] + theme(legend.position='none'),
789                           ncol=1, heights=c(heights[[1]],heights[[2]],heights[[5]]),
790                           widths=c(widths[[1]],widths[[2]],widths[[5]])),
791               arrangeGrob(legend.list[[3]],
792                           plot.list[[3]] + theme(legend.position='none'),
793                           legend.list[[4]],
794                           plot.list[[4]] + theme(legend.position='none'),
795                           ncol=1, heights=c(heights[[3]],heights[[4]]),
796                           widths=c(widths[[3]],widths[[4]])),
797               ncol=2)
798 dev.off()
799
800
801
802 #-------------------------------------------------------------------------------
803 #plotting percent diff where per.diff is > 100%
804
805 plot.list3 <- list()
806 leg.list3 <- list()
807 leg.lim3 <- list()
808
809 leg.pos3 <- list(c(0.5, 0.5), c(0.5, 0.5), c(0.5, 0.5), c(0.5, 0.5), c(0.6, 0.45))
810 leg.width3 <- c(8,8,8,9,10)
811
812 fig.width3 <- c(8.7, 8.5, 9.3, 9.8, 11)
813 fig.height3 <- c(4.3,3.9,8.2,17.5,13.5)
814 fig.prop3 <- list(c(0.3, 0.7), c(0.4, 0.96), c(0.05, 0.95), c(0.05, 0.95), c
        (0.05,0.95))
815
816 for(i in 1:length(hi.list)) {
817   plot.data <- hi.ordered[hi.ordered$variable %in% hi.list[[i]],
818                           c('variable','gentreat','uni.treat','p.val',
819                             'avg','med','diff','per.diff','sig.perdiff')]
820
821   #write.csv(file=paste('check hi data plotted',i,'.csv',sep=''), plot.data)
822
823   #getting legend limits
824   hi.max <- max(plot.data$sig.perdiff[!is.na(plot.data$sig.perdiff)])
825   hi.min <- min(plot.data$sig.perdiff[!is.na(plot.data$sig.perdiff)])
826   hi.larger <- max(hi.max, hi.min)
827   leg.lim3[[i]] <- c(-hi.larger, hi.larger)
828
829   plot.list3[[i]] <- ggplot(plot.data, aes(x=uni.treat, y=variable))
830
831   if(any(is.nan(plot.data$sig.perdiff))==FALSE) {
832     plot.list3[[i]] <- plot.list3[[i]] +
833       geom_tile(aes(fill=sig.perdiff), colour='white')
834     print('here1')
835   }
836
837   else if(any(is.nan(plot.data$sig.perdiff))==TRUE) {
838     plot.list3[[i]] <- plot.list3[[i]] +
839       geom_tile(data=plot.data[!is.na(plot.data$sig.perdiff),],
840                 aes(fill=sig.perdiff),colour='white') +
841       geom_tile(data=plot.data[is.nan(plot.data$sig.perdiff),], colour='white',
842                 fill='grey30',alpha=0.5)
843     print('here2')
844   }
845
846   plot.list3[[i]] <- plot.list3[[i]] +
847     scale_x_discrete(labels=c('Norepinephrine','Repoopulate','Clindamycin+\n\
          tRepoopulate',
848                               'Clindamycin (A)','Clindamycin (B)','Vancomycin')) +
849     scale_fill_gradientn(colours=c('red','white','blue'),
850                          values=rescale(c(-hi.larger,0,hi.larger)),
851                          name='Percent Difference\nin Concentration\t',
852                          limits=leg.lim3[[i]]) +
853     theme_bw(10) +
```

```
854       xlab('Treatment') +
855       ylab('Compound') +
856       theme(panel.grid.major=element_line(colour = 'grey40'),
857             axis.text.x=element_text(hjust=0, vjust=1, angle=-45),
858             legend.position=leg.pos3[[i]], legend.direction='horizontal',
859             legend.key.width=unit(1, 'cm'),
860             legend.key.height=unit(0.3,'cm'),
861             legend.text=element_text(size=7.5),
862             legend.title=element_text(size=7.5))
863
864   leg.list3[[i]] <- g_legend(plot.list3[[i]])
865
866   png(filename=paste('../Plots/treatment effects/per diff-hi range',i,'.png',sep=''
          ),
867       height=fig.height3[i], width=fig.width3[i], units='cm', res=240, pointsize
              =10)
868   fig <- grid.arrange(plot.list3[[i]] + theme(legend.position='none'))
869   dev.off()
870
871   png(filename=paste('../Plots/treatment effects/per diff-hi_leg',i,'.png',sep=''),
872       height=1, width=leg.width3[i], units='cm', res=240, pointsize=10)
873   leg <- grid.arrange(leg.list3[[i]])
874   dev.off()
875 }
876
877 #-------------------------------------------------------------------------------
878 plot.list2 <- list()
879 legend.list2 <- list()
880 fig.list2 <- list()
881 fig.leg2 <- list()
882 leg.lim2 <- list()
883 plot.data.list <- list()
884
885 #customizing plot specifications
886 leg.pos2 <- list(c(0.5, 0.5), c(0.5, 0.5), c(0.5, 0.5), c(0.5, 0.5), c(0.6, 0.45))
887 leg.width2 <- c(8,8,8,9,10)
888
889 fig.width2 <- c(8.7, 8.5, 9.3, 8, 11)
890 fig.height2 <- c(4.3,3.9,8.2,8,13.5)
891 fig.prop2 <- list(c(0.3, 0.7), c(0.4, 0.96), c(0.05, 0.95), c(0.05, 0.95), c
        (0.05,0.95))
892
893 for(i in 1:(length(uni.list2)-1)) {
894   plot.data <- unique(ordered.data2[ordered.data2$variable %in% uni.list2[[i]],])
895
896   plot.data.list[[i]] <- plot.data
897
898   #getting legend limits
899   max.lim <- max(plot.data$diff[!is.nan(plot.data$diff)])
900   min.lim <- min(plot.data$diff[!is.nan(plot.data$diff)])
901   larger.lim <- max(abs(max.lim), abs(min.lim))
902   leg.lim2[[i]] <- c(-larger.lim, larger.lim)
903   print(max.lim)
904   print(min.lim)
905   plot.list2[[i]] <- ggplot(plot.data, aes(x=uni.treat, y=variable))
906
907   if(any(is.na(plot.data$diff))==FALSE) {
908     plot.list2[[i]] <- plot.list2[[i]] +
909       geom_tile(aes(fill=diff), colour='white')
910   }
911
912   else if(any(is.na(plot.data$diff))==TRUE) {
913     plot.list2[[i]] <- plot.list2[[i]] +
914       geom_tile(data=plot.data[!is.nan(plot.data$diff),],
915                 aes(fill=diff),colour='white') +
916       geom_tile(data=plot.data[is.nan(plot.data$diff),], colour='white',
917                 fill='grey30',alpha=0.5)
918
```

```
919     }
920     plot.list2[[i]] <- plot.list2[[i]] +
921       scale_x_discrete(labels=c('Norepinephrine','Repoopulate','Clindamycin+\n\
          tRepoopulate',
922                                 'Clindamycin (A)','Clindamycin (B)','Vancomycin')) +
923       scale_fill_gradientn(colours=c('red','white','blue'),
924                            values=rescale(c(-larger.lim,0,larger.lim)),
925                            name='Difference\nin Concentration\t',
926                            limits=leg.lim2[[i]]) +
927       theme_bw(10) +
928       xlab('Treatment') +
929       ylab('Compound') +
930       theme(panel.grid.major=element_line(colour = 'grey40'),
931             axis.text.x=element_text(hjust=0, vjust=1, angle=-45),
932             legend.position=leg.pos2[[i]], legend.direction='horizontal',
933             legend.key.width=unit(1, 'cm'),
934             legend.key.height=unit(0.3,'cm'),
935             legend.text=element_text(size=7.5),
936             legend.title=element_text(size=7.5))
937
938     legend.list2[[i]] <- g_legend(plot.list2[[i]])
939
940     png(filename=paste('../Plots/treatment effects/diff_bin',i,'.png', sep=''),
941         height=fig.height2[i], width=fig.width2[i], units='cm', res=240, pointsize
              =10)
942     fig.list2[[i]] <- grid.arrange(plot.list2[[i]] + theme(legend.position='none'))
943     dev.off()
944
945     png(filename=paste('../Plots/treatment effects/diff_leg',i,'.png', sep=''),
946         height=1, width=leg.width2[i], units='cm', res=240, pointsize=10)
947     fig.leg2[[i]] <- grid.arrange(legend.list2[[i]], heights=fig.prop2[[i]])
948     dev.off()
949
950 }
```

C.6 Functions

function-summary preprocessing.R

```
1  #////////////////////////////////////////////////////////////////////////////
2  #Function for preprocessing exported files that contain
3  #molecular weight of compounds and pH of samples
4
5  #input: file - is the csv file name, along with its path
6  #       object - is the object name you want to store the processed raw data in
7  #output: data - data frame of the concentration summaries with appropriate column
       names
8  #         and row names
9  #         mol.wt - molecular weights of compounds
10
11 summary.preprocess <- function(file.name){
12
13   #reading csv file
14   data <- read.csv(file=file.name, header=FALSE, check.names=FALSE,
15                    encoding='UTF-8')
16
17   #removing compound molecular weights and sample pH values
18   mol.data <- data[c(3,4), c(1,3:ncol(data))]
19
20   col.name <- as.character(as.matrix((mol.data[1,2:ncol(mol.data)])))
21   row.name <- as.character(as.matrix(mol.data[2,1]))
22
23   mol.weight <- mol.data[2,2:ncol(mol.data)]
24   colnames(mol.weight) <- col.name
25   rownames(mol.weight) <- row.name
```

```
26
27    data <- data[c(3,5:nrow(data)),c(1,3:ncol(data))]
28
29    #creating vector of column names
30    col.name <- as.matrix(cbind('file', data[1,2:ncol(data)]))
31    colnames(data) <- col.name
32
33    #removing redundant row of compounds, compounds are now column names
34    data <- data[2:nrow(data),]
35
36    #turning all concentrations into numeric class
37    for(i in 2:ncol(data)) data[,i] <- as.numeric(as.character(data[,i]))
38
39    #turning all file names into characters
40    data$file <- as.character(data$file)
41
42    return(list('data'=data, 'mol.wt'=mol.weight))
43
44 }
```

function-data cleanup.R

```
1  #////////////////////////////////////////////////////////////////////////////
2  #This function-Removing contaminants and zero-columns
3  data_cleanup = function(dataset, sampleID='sampleID', variable='variable',
4                          value='value', form.y, form.x, contaminants=NULL,
5                          export=FALSE, convert.NA=FALSE) {
6
7    #loading required packages
8    require(reshape2)
9    require(plyr)
10   #Getting list of compounds before anything is removed
11   pre = unique(dataset[,variable])
12
13   #Removing DSS
14   dataset = dataset[!dataset[,variable] %in% c('DSS-d6 (Chemical Shape Indicator)',
15                                                 'DSS (Chemical Shape Indicator)'),]
16
17   check.dss = any(dataset[,variable] %in% c('DSS-d6 (Chemical Shape Indicator)',
18                                             'DSS (Chemical Shape Indicator)'))
19
20   #Removing contaminants
21   dataset = dataset[!dataset[,variable] %in% contaminants,]
22
23   #Getting list of compounds after DSS and contaminants removed
24   post = unique(dataset[,variable])
25
26   #Checking if contaminants were removed
27   check.contaminants = pre[!pre %in% post]
28   check.contaminants = check.contaminants[!check.contaminants %in% c('DSS-d6 (
         Chemical Shape Indicator)',
29                                                                      'DSS (Chemical
                                                                            Shape
                                                                           Indicator)
                                                                           ')]
30
31   #Removing compounds with zero concentrations in every sample
32   subset = ddply(dataset, variable, summarise, sample=sampleID,
33                  value=value)
34
35   zero.values <- ddply(subset, variable, summarise,
36                        unique.values=length(unique(value)))
37
38   constant <- zero.values[zero.values$unique.values == 1,]
39
40   #Removing those rows
41   dataset = dataset[!dataset[,variable] %in% constant[,variable],]
42
```

```
  if(export==TRUE) {
    #Building the formula needed to cast data to wide format
    form.y <- paste(form.y, collapse=' + ')
    form = as.formula(paste(form.y, form.x, sep=' ~ '))

    wide <- dcast(data=dataset, formula=form, value.var=value)
    check2 <- dcast(data=dataset, formula=form, value.var=value)

    write.csv(wide, file='datacheck_wide.csv')
    write.csv(dataset, file='datacheck_long.csv')

  }

  if(convert.NA==TRUE) {
    #converting NAs into 0
    dataset$value[is.na(dataset$value)] <- 0
    write.csv(dataset, file='clean data check.csv')
  }

  #Printing out confirmations
  cat('Compounds with zero or constant concentration in all samples:\n')
  cat(paste('\t', constant[,variable], collapse='\n'),'\n')
  cat('Are there Compounds with na concentration?\n')
  cat(any(is.na(dataset[,value])),'\n')


  cat('DSS included?\n')
  cat(check.dss,'\n')
  cat('Contaminants removed:\n')
  cat(paste('\t', check.contaminants, collpase='\n'),'\n')
  return(dataset)
}
```

function-descriptive statistics.R

```
#////////////////////////////////////////////////////////////////////////////
#Function to obtain descriptive data of data
# calculates mean, standard deviation, standard error, ymax, ymin
# has option to scale data, and correspoding descriptive stats

#input: full.data - qualifier and concentration data
#       data.type - 'long' or 'wide'; specifies format of full.data
#       varaible - character string of how you want long data to be broken up;
#                  not required for wide data
#       qualifier - qualifier data; used so can separate out from full.data
#       scale.data - logical; retrieve stats on the scaled version of the data;
#                    mean centres and unit variance scales the data

#output: stat - data frame of descriptive statistics
#------------------------------------------------------------------------------>
descriptive.stat <- function(full.data, data.type='wide', variable=NULL,
                             qualifier=NULL, scale.data=FALSE) {

  require(plyr)
  if(data.type=='wide') {
    #separating concentration data from qualifier data in full.data
    data <- full.data[,(ncol(qualifier)+1):ncol(full.data)]

    if(scale.data==FALSE) {
      stat <- data.frame(avg = apply(data, 2, mean),
                         med = apply(data, 2, median),
                         stdev = apply(data, 2, sd),
                         compound = colnames(data))

      stat$se <- stat$stdev/sqrt(nrow(data))
      stat$ymax <- stat$avg + stat$se
      stat$ymin <- stat$avg - stat$se
    }
```

```
34
35      else if(scale.data==TRUE){
36        stat <- data.frame(scale.avg = apply(data, 2, mean),
37                           scale.med = apply(data, 2, med),
38                           scale.stdev = apply(data, 2, sd),
39                           compound = colnames(data))
40
41        stat$scale.se <- stat$scale.stdev/sqrt(nrow(data))
42        stat$scale.ymax <- stat$scale.avg + stat$scale.se
43        stat$scale.ymin <- stat$scale.avg - stat$scale.se
44
45      }
46    }
47
48    else if(data.type=='long') {
49      if(scale.data==FALSE) {
50        stat <- ddply(full.data, variable, mutate,
51                      avg = mean(value),
52                      med = median(value[value != 0]),
53                      stdev = sd(value),
54                      n=length(value))
55
56        stat$se <- stat$stdev/sqrt(stat$n)
57        stat$ymax <- stat$avg + stat$se
58        stat$ymin <- stat$avg - stat$se
59
60        #turning all NAs to zero in median
61        stat$med[is.na(stat$med)] <- 0
62
63      }
64      else if(scale.data==TRUE) {
65        stat <- ddply(full.data, variable, mutate,
66                      scale.avg = mean(value),
67                      scale.med = median(value[value != 0]),
68                      scale.stdev = sd(value))
69
70        stat$scale.se <- stat$scale.stdev/sqrt(nrow(full.data))
71        stat$scale.ymax <- stat$scale.avg + stat$scale.se
72        stat$scale.ymin <- stat$scale.avg - stat$scale.se
73      }
74    }
75    return(stat)
76  }
```

function-extract legend.R

```
1  #////////////////////////////////////////////////////////////////////////////////
       //////
2  #function to extract legend from a ggplot
3  #to be used the grid.arrrange from gridExtra package
4
5  #Extract Legend
6  g_legend<-function(a.gplot){
7    tmp <- ggplot_gtable(ggplot_build(a.gplot))
8    leg <- which(sapply(tmp$grobs, function(x) x$name) == "guide-box")
9    legend <- tmp$grobs[[leg]]
10   return(legend)}
```

function-pca.R

```
1  #////////////////////////////////////////////////////////////////////////
2
3  #Functions for making processing and analyzing data subsets created earlier
4
5  #------------------------------------------------------------------------>
6  #Function for Performing PCA analysis
7    #when running the script:
```

```
8   #enter as: pca_processed_data --> process to be applied to the output_numeric
        data, PCA
9
10
11  #input: data - numeric data to be used for PCA, called output_numeric, unpackaged
        from 'function-data input' file
12  #       form.y - y variable in formula; usually is qualifier column names to be
        put into formula that casts data
13  #       form.x - x variable in formula; usually the variables involved (i.e.
        compound)
14  #       variable - the column name in which variables are stored (i.e. 'Compound
        ')
15  #       value - the column name in which data values are stored (i.e. '
        concentration')
16  #       scale - 'UV' for unit variance, 'UVN' for no unit variance, 'pareto'
17  #       center - mean centering; default to TRUE
18  #output: PCA-processed data, called pca_data
19 #------------------------------------------------------------------------------>
20 #Loading required packages
21 require(reshape2)
22
23 perform_pca = function(data, sampleID='sample_name', variable, value,
24                        form.y, form.x, scale, center = TRUE, convert.NA=FALSE) {
25
26
27
28  #Getting column names for qualifiers and for concentrations
29  numeric.long = c(variable,value)
30  qual.long = colnames(data)[!colnames(data) %in% numeric.long]
31
32  #Building the formula needed to cast data to wide format
33  form.y <- paste(form.y, collapse=' + ')
34  form = as.formula(paste(form.y, form.x, sep=' ~ '))
35
36  #Casting data to wide format; includes qualifiers
37  data.wide = dcast(data=data, formula=form,
38                    value.var=value)
39
40  if(convert.NA==TRUE) {
41    #converting NAs into 0
42    data.wide[is.na(data.wide)] <- 0
43    write.csv(data.wide, file='data_check2.csv')
44  }
45
46  #separating out concentration and qualifiers in wide format
47  conc.wide = data.wide[, unique(data[,variable])]
48  qual.wide = data.wide[, qual.long]
49
50  #Converting concentration data to numeric class
51  for(i in conc.wide) { i <- as.numeric(i)}
52
53  #Unit Variance Scaling, scaling weight of 1/sk
54  if(scale == 'UV') {
55    scale = TRUE
56  }
57  #No Unit Variance Scaling
58  else if(scale == 'UVN') {
59    scale = FALSE
60  }
61  #Pareto scaling, scaling weight of 1/(sk)e-2
62  else if(scale == 'pareto') {
63    scale =  for(column in 1:ncol(data)) {
64      data[,column] = data[,column]/sqrt(sd(data[,column]))
65    }
66  }
67
68  #Creating pca model
69  pca_data = prcomp(conc.wide, scale=scale, center=center,
```

```
                        na.action=na.omit)

  print(summary(pca_data))

  #Adding qualifiers to pca score data
  pca_data$x = cbind(qual.wide, pca_data$x)

  #Melting pca score data back to long format,
  #where variable is PC number and value is score value on that PC
  pca.melt = melt(pca_data$x, id.vars=qual.long)

  #Removing 'PC' in PC column names, so only left with a number
  pca.melt$variable = as.numeric(gsub('PC', '', pca.melt$variable))

  #Get rid of last PC if it is odd
  if(max(pca.melt$variable) %% 2 == 1) {
    pca.melt = pca.melt[pca.melt$variable != max(pca.melt$variable), ]
  }

  #Creating a new object where has columns of of odd number PCs and even number PCs
  odd.PCs = pca.melt$variable %% 2 == 1
  split = pca.melt[odd.PCs, qual.long]
  split$x.value = pca.melt$value[odd.PCs]
  split$y.value = pca.melt$value[!odd.PCs]
  #Creating a column called PC, to give the PC number
  split$x.PC = pca.melt$variable[odd.PCs]
  split$y.PC = pca.melt$variable[!odd.PCs]
  #Adding column 'view' to describe which PCs are being plotted
  split$view = paste('PC',pca.melt$variable[odd.PCs], 'vs',
                     'PC',pca.melt$variable[!odd.PCs])

  #Replacing the pca model score data with the melted and sorted version created
      here
  pca_data$x = split

  return(pca_data)
  }
```

function-pca diagnostics.R

```
#////////////////////////////////////////////////////////////////////////

#Functions for performing diagnostic tests on the pca data, evaluating which PCs to
     consider

#------------------------------------------------------------------------>

#Function for performing PCA diagnostics
  #when running the script:
  #enter as = diagnostic_results
  #unpackage as diagnostic_results_number = diagnostics_data[['number']]
               # diagnostics_results_explained = diagnostics_data[['explained']]
              # diagnostics_results_plot = diagnostics_data[['plot']]

  #input: the PCA model output of prcomp()
  #Output: varies, defined by individual function

#------------------------------------------------------------------------>
#Scree plot diagnostic
#
#Output: 'number' - the number of PCA components to use
#   'explained' - % variance explained by 'number' PCs
#  'plot'   - ggplot object with generated scree plot

scree_plot = function(model)
{
  #Making sure the right packages are loaded
  require(ggplot2)
```

```
28
29    #Testing if the model is indeed the result of a prcomp function
30    if (class(model) != 'prcomp')
31    {
32      stop('Input to scree_plot must be the output of prcomp() function')
33    }
34
35    #Converting standard deviations to variance
36    variance = model$sdev**2
37    variance = variance/sum(variance)*100
38
39    #Looking at differences between sequential variance values
40    difference = variance[-length(variance)] - variance[2:length(variance)]
41
42    #Creating a permanent dataset that will be used later for plotting
43    #for now, every value is assigned INCLUDE, and INCLUDE is defined as TRUE
44    dataset = data.frame(x=1:length(difference),
45                         y=difference,
46                         INCLUDE=TRUE)
47
48    #Looping through data to exclude PC variance differences, and creating a straight
          line fit
49    #with every iteration
50    for (i in 1:min(10, nrow(dataset)-2))
51
52    {
53      #Marking point for exclusion, making all INCLUDE values equal to FALSE
54      dataset[i, 'INCLUDE'] = FALSE
55
56      #Calculating length of x vector (in sum, TRUE is equal to 1, FALSE, 0)
57      x_length = sum(dataset$INCLUDE)
58
59      #Generating weights for linear regression. We want the first points to have a
            lot
60      #more weight than the last points. To do this, we create a weight vector from 1
            to
61      #0.1 in equal segments, then cube it to amplify the effect.
62      weight = seq(1, .1, length.out=x_length)**3
63
64      #Fitting linear model, lm = to fit linear model, y ~ x means y=f(x), where y is
            the difference in variance between two successive values
65      #fitting the linear model with the dataset assigned previously, and using all
            rows with INCLUDE in its column
66      model = lm(y ~ x, data=dataset[dataset$INCLUDE,])
67
68      #Extracting probability that the slope is actually 0
69      prob = summary(model)$coefficients[2,4]
70
71      #see objects of the model by >print(summary(model)).
72      #Column 2 of object Coefficients is the Std. Error
73      #Column 4 of object Coefficients is Pr(>|t|), which in context, gives the
            probability that the slope is zero
74
75      #If the probability is sufficiently high, the slope
76      #is deemed to be zero and there is limited improvement
77      #from subsequent PC components; exit script
78      if (prob > 0.05) break
79    }
80
81    #Storing the last PC index (i) before the loop terminated
82    n = i
83
84    #Calculating how much variance this number
85    #of components will explain
86    explained = sum(variance[1:n])
87
88    #Creating a new variance dataset from diagnostic results
89    diagnostic_results = data.frame(x=1:length(variance),
```

```
                        y=variance,
                        Type=ifelse(c(dataset$INCLUDE, TRUE), #test if INCLUDE in
                            the dataset is TRUE
                                'Not significant', #if yes, type out 'Not
                                    significant'
                                'Significant'))    #if no, type out "Significant
                                    "

  #Creating a basic ggplot object as validation
  p = ggplot(diagnostic_results)
  p = p + geom_point(aes(x=x, y=y, colour=Type))
  p = p + stat_smooth(data=diagnostic_results[diagnostic_results$Type == 'Not
      significant',],
                      aes(x=x, y=y), se=FALSE, method='lm', colour='black')

  p = p + xlab('Principal Component')
  p = p + ylab('Variance explained (%)')
  p = p + scale_colour_manual(values=c('Significant'='red', 'Not significant'='
      black'))

  return(list('number'=n, 'explained'=explained, 'plot'=p))
}
```

function-plotting D score.R

```
#////////////////////////////////////////////////////////////////////////////

#Functions for plotting pca results

#List of defined functions:
# score_plot_2d

#Making sure the right packages are loaded
require(ggplot2)
require(grid)
require(plyr)
require(reshape2)
require(ellipse)

#sourcing required functions
source('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Functions/
    function-plotting 1D score.R')

#------------------------------------------------------------------------------>

#Function for plotting 2D score plots
#when running the script:
#enter as: score_plot

#Input: the PCA model output of prcomp()
#Output: A single ggplot2 plot object

#Note: Here, the use of "2D" implies that 2 principal components
#are considered at a time. As the plotting is actually
#quite simple, this is no more than a shortcut function for
#repeated use. Two different modes are used. In 'full' mode,
#each principal component is compared against each other i.e.
#PC 1 vs PC 2, PC 1 vs PC 3, etc... In 'compressed' mode, only
#successive principal components are compared i.e. PC 1 vs PC 2,
#PC 3 vs PC 4, etc...
#------------------------------------------------------------------------------>

#Input: 'pca_data' - the PCA model output of prcomp()
      # 'PCs' - vector of Principal Components to consider
      # 'colour' - name of qualifer_data column to be used as the colour aesthetic
      # 'shape' - name of qualifer_data column to be used as the shape aesthetic
      # 'size' - name of qualifer_data column to be used as the size aesthetic
      # 'label' - name of qualifer_data column to be used as the point labels
```

```
43        # 'view_mode' - the view at which the pricipal components are compared;
              either compressed or full,
44        #              where compressed is PC1 v PC2, PC3 v PC4 etc, and full is PC1 v
               PC2, PC1 v PC3, PC1 v PC4 etc.
45        # 'black_white' - plot will be in black and white
46  #Output: 'plot'   - ggplot object with generated plot
47
48  #------------------------------------------------------------------------>
49
50  score_plot_2d = function(pca_data, PCs=1:4,
51                           colour=NULL, shape=NULL, size=NULL, label=NULL, title=NULL
                                 ,
52                           legend.ori=NULL, outline=FALSE, black_white=FALSE)
53  {
54    #------------------------------------------------------>
55    #Input testing
56
57    #Testing if the pca_data is indeed the result of a prcomp function
58    if (class(pca_data) != 'prcomp')
59    {
60      stop('Input "pca_data" to score_plot_2d must be the output of prcomp() function
            ')
61    }
62
63    #Testing qualifier rownames to make sure they are the same as the rownames of the
          score matrix
64    #if (length(intersect(rownames(pca_data$x), rownames(qualifier_data))) != length(
          rownames(qualifier_data)))
65    #{
66    #  stop('Input "qualifier_data" to score_plot_2d must have the same rownames as
          the score matrix in "pca_data"')
67    #}
68
69    #Testing if PCs are numeric
70    if (!is.numeric(PCs))
71    {
72      stop('Input "PCs" to score_plot_2d must be a vector of numeric values
            corresponding to target principal components')
73    }
74
75    #Testing to make sure that PCs are in the input data
76    if (any(PCs > ncol(pca_data$x)))
77    {
78      stop('Input "PCs" to score_plot_2d must not be outside of the pca_data
            principal component range')
79    }
80
81    #Checking if colour, shape, or size conditions are column names in qualifer_data
82    aesthetics = c('colour'=colour, 'shape'=shape, 'size'=size)
83    aesthetics = aesthetics[!is.null(aesthetics)] #Excluding default NULL entries
84    #------------------------------------------------------>
85
86    #Extract scores
87    score.data = as.data.frame(pca_data$x)
88
89    #Copying out the standard deviation info from pca_data as a dataframe
90    variance = as.data.frame(pca_data$sdev)
91    #Converting standard deviations to variance
92    variance = variance**2
93    variance = variance/sum(variance)*100
94    variance = round(variance, digits=1)
95
96    #Initializing a new vector to store collected.data
97    collected.data = c()
98
99    #The way in which the collected data is built up and
100   #If only 1 PC was specified, calling
101   #score_plot_1d instead
```

```
102   if (length(PCs) == 1)
103   {
104     p = score_plot_1d(pca_data, PCs,
105                       colour, shape, size, label, pt_lab, title,
106                        perform_checks=FALSE, group_labels)
107      return(p)
108   }
109
110   #Iterating through the PCs, two at a time.
111   for (i in seq(1, length(PCs), by=2))
112   {
113     view.data = score.data[score.data$x.PC==i,]
114     view.data$view.var = paste('PC', i, '(', variance[i], '%) ','vs.',
115                                'PC', i+1,'(', variance[i+1], '%)')
116
117     #Adding to collected.data
118     collected.data = rbind(collected.data, view.data)
119
120   }
121
122   #Converting View to factor, preserving ordering
123   collected.data$view.var = factor(collected.data$view.var,
124                                    levels=unique(collected.data$view.var))
125
126   #Calculating plot limits
127   xlimit = max(abs(collected.data$x.value))*1.15
128   ylimit = max(abs(collected.data$y.value))*1.1
129
130   #---------------------------------------------------------
131   #Generating plot
132
133   p = ggplot(collected.data)
134
135   #If size descriptor has been specified, use it as an aesthetic,
136    if(!is.null(size)) {
137     p = p + geom_point(aes_string(x='x.value', y='y.value',
138                           colour=colour, shape=shape, size=size), alpha=0.8)
139   }
140   else
141   { #otherwise, default to 3 and make points with black outline
142     if(outline==TRUE){
143       p = p + geom_point(aes_string(x='x.value', y='y.value', shape=shape),
144                          colour='black', size=4, alpha=0.8)
145     }
146
147     p = p + geom_point(aes_string(x='x.value', y='y.value', colour=colour, shape=
          shape),
148                        size=3, alpha=0.8)
149     #Alternative script for making points with black outline
150     #though there is internal bug with legend
151     #p = p + geom_point(aes_string(x='xPC', y='yPC', fill=colour, shape=shape),
152     #                   colour='black', size=4, alpha=0.8)
153     #p = p + scale_shape_manual(values=21:25)
154   }
155
156   #Cusomizing colours of points
157   #if colour is discrete give customized colours
158   if(!is.numeric(collected.data[,colour]) & length(unique(collected.data[,colour]))
        < 8) {
159     p = p + scale_colour_manual(values=c('deeppink1','darkorange1','cornflowerblue'
          ,
160                                          'darkorchid4','chartreuse','blue', 'grey30
                                               '))
161
162   }
163   else if(!is.numeric(collected.data[,colour]) & length(unique(collected.data[,
        colour])) >= 8 ) {
164     p = p + scale_colour_brewer(palette='Dark2')
```

```
165
166   }
167   #if colour is continuous give colour gradiaent
168   else if(is.numeric(collected.data[,colour])) {
169
170     p = p + scale_colour_gradient(low='red')
171   }
172
173
174   #Adding text only if the "label" descriptor has been provided
175   if (!is.null(label))
176   {
177     if (label == 'Plot_Treat') {
178       p = p + geom_text(aes_string(x='x.value', y='y.value', label=label, alpha='
              Plot_Treat'),
179                         hjust=-0.05, vjust=0.75,
180                         size=5, fontface='plain', lineheight=0.8)
181       p = p + scale_alpha_manual(values=list('None'=0, 'Norepinephrine'=1,
182                                              'Clindamycin'=1, 'Repoopulate'=1,
183                                              'Clindamycin+RePOOPulation'=1), guide=
                                                  FALSE)
184     }
185     else {
186       p = p + geom_text(aes_string(x='x.value', y='y.value', label=label),
187                         hjust=-0.5, vjust=0.3,
188                         size=5, fontface='plain', lineheight=0.8)
189
190     }
191   }
192
193   #Creating  scheme for black and white plots
194   if (black_white == TRUE){
195     p = p + geom_point(aes_string(x='x.value', y='y.value', shape=shape, fill=
              colour),
196                        size=3, alpha=0.8) +
197       scale_shape_manual(values=21:25) +
198       scale_fill_manual(values=c('grey30', 'white'))
199   }
200
201   #Adding title only if "title" descriptor has been provided
202   if (!is.null(title)) p = p + ggtitle(label=title)
203
204   #Drawing origin
205   p = p + geom_vline(xintercept=0, alpha=0.3)
206   p = p + geom_hline(yintercept=0, alpha=0.3)
207
208   p = p + xlab('PC score') +
209     ylab('PC score') +
210     theme_bw(16) +
211     theme(axis.title.x = element_text(size=14, face='bold'),
212           axis.title.y = element_text(size=14, face='bold'),
213           axis.text = element_text(size=12, colour='black'),
214           legend.title = element_text(size=14),
215           legend.text = element_text(size=14),
216           legend.key = element_rect(colour='white'),
217           legend.margin = unit(0, "cm"),
218           legend.position = 'bottom',
219           panel.border=element_rect(size=2, colour='black'),
220           panel.margin=unit(.05, 'npc'))
221
222   #Legend orientation
223   if (legend.ori == 'horizontal') {
224     p = p + theme(legend.direction='horizontal',
225                   legend.position='bottom')
226   }
227   else if(legend.ori == 'vertical') {
228     p = p + theme(legend.direction='vertical',
229                   legend.position='right')
```

```
  }

  #Setting limits
  p = p + xlim(-xlimit,xlimit)
  p = p + ylim(-ylimit,ylimit)

  #While the rest of the plotting code is generic,
  #the facetting will change based on view
  #if (view_mode == 'compressed')
  p = p + facet_wrap(~ view.var, ncol=min(floor(length(PCs)/2), 3)) +
  theme(strip.text.x = element_text(size=12, face='bold'))
  #else if (view_mode == 'full') p = p + facet_grid(yLabel ~ xLabel)

  return(list('collected.data'=collected.data, 'p'=p))
}
```

function-plotting D loading.R

```
#////////////////////////////////////////////////////////////////////////
#Function for plotting2D Loading plot
#
#Here, the use of "2D" implies that 2 principal components
#are considered at a time. Unlike the score plot, loading plot groupings
#(pathways) are not exclusive i.e. a compound may belong to more than one
#grouping. This makes it difficult to use colour, shape, and other aesthetics
#in the same manner as for the score plot. For now, these options will be
#removed
#
#when running the script:
#enter as: loading_plot
#
#Input: 'pca_data' - the PCA model output of prcomp()
#  'PCs' - vector of Principal Components to consider
#  'label' - vector of compounds that should be labelled on the score plot
# 'view_mode' - one of either 'full' or 'compressed' (see score plot)
#
#Output: 'plot' - ggplot object with generated plot

loading_plot_2d = function(pca_data, PCs=1:4, label=NULL, outline=FALSE,
                           title=NULL, view_mode='compressed')
{
  #------------------------------------------------------------>
  #Input testing

  #Testing if the pca_data is indeed the result of
  #a prcomp function
  if (class(pca_data) != 'prcomp')
  {
    stop('Input "pca_data" to score_plot_2d must be the output of prcomp() function
        ')
  }

  #Testing if PCs are numeric
  if (!is.numeric(PCs))
  {
    stop('Input "PCs" to score_plot_2d must be a vector of numeric values
        corresponding to target principal components')
  }

  #Testing to make sure that PCs are in the input data
  if (any(PCs > ncol(pca_data$rotation)))
  {
    stop('Input "PCs" to score_plot_2d must not be outside of the pca_data
        principal component range')
  }

  #Checking if given labels are in the loading data
  if (any(!label %in% rownames(pca_data$rotation)))
```

```
48  {
49    stop(paste('The following labels do not correspond to variables in "pca_data":\
        n\t',
50                 paste(label[!label %in% rownames(pca_data$rotation)], collapse=', ')
                       ))
51  }
52  #------------------------------------------------------------>
53
54  #Extract loadings
55  loading_data = as.data.frame(pca_data$rotation)
56
57  variance = as.data.frame(pca_data$sdev)
58  #Converting standard deviations to variance
59  variance = variance**2
60  variance = variance/sum(variance)*100
61  variance = round(variance, digits=1)
62
63  #Initializing a new vector to store collected_data
64  collected_data = c()
65
66  #The way in which the collected data is built up and
67  #presented depends on the view_model
68  if (view_mode == 'compressed')
69  {
70
71    #If only 1 PC was specified, calling
72    #loding_plot_1d instead
73    if (length(PCs) == 1)
74    {
75      p = loading_plot_1d(pca_data, PCs, label, title,
76                          perform_checks=FALSE)
77      return(p)
78    }
79    #Iterating through the PCs, two at a time.
80    #If there is an odd number of PCs, the last one is dropped
81    for (i in seq(1, length(PCs)-1, by=2))
82    {
83      view_data = loading_data[, c(i, i+1)]
84      view_data$View = paste('PC', i, '(', variance[i], '%) ','vs.',
85                             'PC', i+1,'(', variance[i+1], '%)')
86
87      #Changing score_column names
88      colnames(view_data)[1:2] = c('xPC', 'yPC')
89
90      #Initializing text column with blanks
91      view_data$Text = ''
92
93      #Adding labels
94      view_data[label, 'Text'] = label
95
96      #Adding to collected_data
97      collected_data = rbind(collected_data, view_data)
98    }
99  }
100 else if (view_mode == 'full')
101 {
102   #Iterating through the PCs one at a time. As the comparison
103   #is factorial, two loops are used, 1v2, 1v3, ..., 2v1, 2v2, etc
104   for (i in PCs)
105   {
106     for (j in PCs)
107     {
108       view_data = loading_data[, c(i, j)]
109       view_data$xLabel = paste('PC', i,'(', variance[i], '%)')
110       view_data$yLabel = paste('PC', j, '(', variance[j], '%)')
111
112       #Changing score_column names
113       colnames(view_data)[1:2] = c('xPC', 'yPC')
```

```
114
115         #Initializing text column with blanks
116         view_data$Text = ''
117
118         #Adding labels
119         view_data[label, 'Text'] = label
120
121         #Adding to collected_data
122         collected_data = rbind(collected_data, view_data)
123       }
124     }
125   }
126   else
127   {
128     stop('Input "view_mode" must be one of "full" or "compressed"')
129   }
130
131   #Calculating plot limits
132   xlimit = max(abs(collected_data$xPC))*1.1
133   ylimit = max(abs(collected_data$yPC))*1.1
134
135   #Generating plot
136   p = ggplot(collected_data)
137
138   #If highlighting will be performed, use a less stark, grey tone,
139   #otherwise, use black with transparency to limit overplotting
140   if (is.null(label)) p = p + geom_point(aes(x=xPC, y=yPC), colour='black', size=5,
          alpha=.7)
141   else if(is.null(label) & outline == TRUE) {
142     p = p + geom_point(aes(x=xPC, y=yPC), colour='grey', size=5)
143   }
144   #Adding text only if the "label" descriptor has been provided,
145   #the points with text are also highlighted
146   if (!is.null(label))
147   {
148     p = p + geom_text(aes(x=xPC, y=yPC, label=Text),
149                       hjust=-0.1, vjust=0, size=5, fontface=2)
150
151   }
152   if(!is.null(label) & outline==TRUE) {
153     p = p + geom_point(data=collected_data[collected_data$Text != '',],
154                        aes(x=xPC, y=yPC), size=5, shape=1)
155   }
156
157   #Adding title only if "title" descriptor has been provided
158   if (!is.null(title)) p = p + ggtitle(label=title)
159
160   #Drawing origin
161   p = p + geom_vline(xintercept=0, alpha=0.3)
162   p = p + geom_hline(yintercept=0, alpha=0.3)
163
164   p = p + xlab('PC loading') + ylab('PC loading')
165
166   #Setting limits
167   p = p + xlim(-xlimit,xlimit)
168   p = p + ylim(-ylimit,ylimit)
169
170   #While the rest of the plotting code is generic,
171   #the facetting will change based on view
172   if (view_mode == 'compressed') p = p + facet_wrap(~ View, ncol=min(floor(length(
          PCs)/2), 3)) +
173     theme_bw(16) +
174     theme(axis.title.x = element_text(size=14, face='bold'),
175           axis.title.y = element_text(size=14, face='bold'),
176           axis.text = element_text(size=12, colour='black'),
177           legend.title = element_text(size=14),
178           legend.text = element_text(size=12),
179           legend.key = element_rect(colour='white'),
```

```
180             legend.margin = unit(0, "cm"),
181             panel.border=element_rect(size=2, colour='black'),
182             strip.text.x = element_text(size=14, face='bold'))
183   else if (view_mode == 'full') p = p + facet_grid(yLabel ~ xLabel) +
184     theme_bw(16) +
185     theme(axis.title.x = element_text(size=14, face='bold'),
186           axis.title.y = element_text(size=14, face='bold'),
187           axis.text = element_text(size=12, colour='black'),
188           legend.title = element_text(size=14),
189           legend.text = element_text(size=12),
190           legend.key = element_rect(colour='white'),
191           legend.margin = unit(0, "cm"),
192           strip.text.x = element_text(size=14, face='bold'))
193
194   return(list('p'=p, 'collected.data'=collected_data))
195 }
```

function-scaling.R

```
1  #////////////////////////////////////////////////////////////////////////////
   /
2  #This function is to scale metabolite data before it goes to PCA processing,
3  #or without PCA processeing:
4  #     Scaling to Acetate
5  #     Mean centering
6  #     Unit variance scaling
7  # Note: mean centering and unit variance is done through the perform_pca function,
8  #       so if passing data from
9
10 #input: data - unprocessed concentration data
11 #       normalize - scale all compounds within one observation to a single compound
   ,
12 #                   divides every compound by the concentration of normalizing
   compound
13 #
14 #output: scaledData - concentration data that has undergone the specified scaling
15 #
16 #////////////////////////////////////////////////////////////////////////////
   /
17
18 require(reshape2)
19
20
21
22 scaling = function(data, sampleID='sampleID', variable='variable', value='value',
23                    normalize=NULL, centre=TRUE, UV=FALSE) {
24
25   if(!is.null(normalize)){
26     setwd('C:/Users/Sandi/Dropbox/Liquid Gold/R Scripts/V 3.0/')
27     source('./Functions/function-normalize.R')
28     normalized = ddply(data, sampleID, f_normalize, variable=variable, scal=value,
29                        normalize=normalize)
30
31     #remove the compound by which the concentrations are scaled, otherwise PCA will
         fail
32     processed.data = normalized[normalized[,variable] != normalize, ]
33     print('data is normalized by Acetate')
34     print('Acetate removed from analysis as its value is one')
35   }
36   if(is.null(normalize)){
37     processed.data = data
38   }
39
40   if(centre==TRUE){
41     processed.data = ddply(processed.data, variable, mutate,
42                            scal=value-mean(value))
43     print('data is mean centred')
44   }
```

```
  if(centre==FALSE){
    processed.data = processed.data
  }

  if(UV==TRUE){
    processed.data = ddply(processed.data, variable , mutate,
                           scal=scal/sd(scal))
    print('data is scaled by unit variance')
  }
  if(UV==FALSE){
    processed.data = processed.data
  }

  return(processed.data)
}
```

function-t test.R

```
#////////////////////////////////////////////////////////////////////////////////////////////////
#function to perform the t test

#First getting the t statistic for each compound
tStatistic = function(data, value, variable, group1, group2, n) {

  #Separating out population and test groups
  pop = data[data[, variable] == group1,]
  pop = pop[,colnames(pop) == value]

  sample = data[data[, variable] == group2,]
  sample = sample[,colnames(sample) == value]

  #Calculating t statistic
  pop = ddply(pop, 'compound', transform, average = mean(concentration))
  pop = ddply(pop, 'compound', mutate, stdev = sd(concentration))
  pop = pop[,colnames(pop)%in% c('compound', 'average', 'stdev')]
  pop = unique(pop)
  print(head(pop))

  if(FALSE) {
  sample = ddply(sample, 'compound', transform, average = mean(concentration))
  sample = sample[,colnames(sample) %in% c('compound', 'average')]

  calc1 = data.frame(compound = pop1.calc$compound,
                     sub = abs(pop1.calc$average-pop2.calc$average))
  calc2 = data.frame(compound = control.calc$compound,
                     ss = control.calc$stdev*sqrt(2/n))
  ss = data.frame(compound = pop1.calc$compound, ss = calc1$sub/calc2$ss)
  }

  return(list('pop'=pop, 'sample'=sample,
              'calc1'=calc1, 'calc2'=calc2, 'ss'=ss))
  #abs(apply(control,2,mean)-apply(treatment,2,mean)/(apply(control, 2, sd)*sqrt(2/
      n)))
}

if(FALSE) {
test = tStatistic(comp.input, value='composition', variable='Gen_Treat',
                  group1='None', group2='Clindamycin', 2)

control.calc = test[['pop1.calc']]
treat.calc = test[['pop2.calc']]
calc1 = test[['calc1']]
calc2 = test[['calc2']]
ss = test[['ss']]
}
if(FALSE){
  t_test_func = function(data, value, variable, group1, group2) {
```

```
    x = data[data[, variable] == group1,]
    x = x[,colnames(x) == value]

    y = data[data[, variable] == group2,]
    y = y[,colnames(y) == value]

    test = t.test(x, y)
    out = test$p.value
    return(out)
  }
}
```

function-siegel tukey test.R

```
siegel.tukey <- function(x, y, id.col = FALSE, adjust.median = F,
                         rnd = -1, alternative = "two.sided", mu = 0, paired =
                             FALSE,
                         exact = FALSE, correct = TRUE, conf.int = FALSE, conf.
                             level = 0.95) {
  ###### published on:
  #   http://www.r-statistics.com/2010/02/siegel-tukey-a-non-parametric-test-for-
      equality-in-variability-r-code/
  ## Main author of the function:  Daniel Malter

  # x: a vector of data

  # y: Group indicator (if id.col=TRUE); data of the second
  #   group (if
  # id.col=FALSE). If y is the group indicator it MUST take 0
  #   or 1 to indicate
  # the groups, and x must contain the data for both groups.

  # id.col: If TRUE (default), then x is the data column and y
  #   is the ID column,
  # indicating the groups. If FALSE, x and y are both data
  #   columns. id.col must
  # be FALSE only if both data columns are of the same length.

  # adjust.median: Should between-group differences in medians
  #   be leveled before
  # performing the test? In certain cases, the Siegel-Tukey
  #   test is susceptible
  # to median differences and may indicate significant
  #   differences in
  # variability that, in reality, stem from differences in
  #   medians.

  # rnd: Should the data be rounded and, if so, to which
  #   decimal? The default
  # (-1) uses the data as is. Otherwise, rnd must be a
  #   non-negative integer.
  # Typically, this option is not needed. However,
  #   occasionally, differences in
  # the precision with which certain functions return values
  #   cause the merging
  # of two data frames to fail within the siegel.tukey
  #   function. Only then
  # rounding is necessary. This operation should not be
  #   performed if it affects
  # the ranks of observations.

  # ... arguments passed on to the Wilcoxon test. See
  #   ?wilcox.test

  # Value: Among other output, the function returns the data,
  #   the Siegel-Tukey
  # ranks, the associated Wilcoxon's W and the p-value for a
  #   Wilcoxon test on
```

```
52  # tie-adjusted Siegel-Tukey ranks (i.e., it performs and
53  #   returns a
54  # Siegel-Tukey test). If significant, the group with the
55  #   smaller rank sum has
56  # greater variability.
57
58  # References: Sidney Siegel and John Wilder Tukey (1960) "A
59  #   nonparametric sum
60  # of ranks procedure for relative spread in unpaired
61  #   samples." Journal of the
62  # American Statistical Association. See also, David J.
63  #   Sheskin (2004)
64  # "Handbook of parametric and nonparametric statistical
65  #   procedures." 3rd
66  # edition. Chapman and Hall/CRC. Boca Raton, FL.
67
68  # Notes: The Siegel-Tukey test has relatively low power and
69  #   may, under certain
70  # conditions, indicate significance due to differences in
71  #   medians rather than
72  # differences in variabilities (consider using the argument
73  #   adjust.median).
74
75  # Output (in this order)
76
77  # 1. Group medians (after median adjustment if specified)
78  # 2. Wilcoxon-test for between-group differences in medians
79  #   (after the median
80  # adjustment if specified)
81  # 3. Data, group membership, and the Siegel-Tukey ranks
82  # 4. Mean Siegel-Tukey rank by group (smaller values indicate
83  #   greater
84  # variability)
85  # 5. Siegel-Tukey test (Wilcoxon test on tie-adjusted
86  #   Siegel-Tukey ranks)
87
88  is.formula <- function(x) class(x) == "formula"
89
90  if (is.formula(x)) {
91    y <- do.call(c, list(as.name(all.vars(x)[2])), envir = parent.frame(2))
92    x <- do.call(c, list(as.name(all.vars(x)[1])), envir = parent.frame(2))  # I am
          using parent.frame(2) since if the name of the variable in the equation is
          'x', then we will mistakenly get the function in here instead of the
          vector.
93    id.col <- TRUE
94    # print(x)
95    # print(ls.str())
96    # data=data.frame(c(x,y),rep(c(0,1),c(length(x),length(y))))
97    # print(data)
98  }
99
100 if (id.col == FALSE) {
101   data = data.frame(c(x, y), rep(c(0, 1), c(length(x), length(y))))
102 } else {
103   data = data.frame(x, y)
104 }
105 names(data) = c("x", "y")
106 data = data[order(data$x), ]
107 if (rnd > -1) {
108   data$x = round(data$x, rnd)
109 }
110
111 if (adjust.median == T) {
112   ##cat("\n", "Adjusting medians...", "\n", sep = "")
113   data$x[data$y == 0] = data$x[data$y == 0] - (median(data$x[data$y ==
114                                                              0]))
115   data$x[data$y == 1] = data$x[data$y == 1] - (median(data$x[data$y ==
116                                                              1]))
```

```
117   }
118   ##cat("\n", "Median of group 1 = ", median(data$x[data$y == 0]),
119   ##    "\n", sep = "")
120   ##cat("Median of group 2 = ", median(data$x[data$y == 1]), "\n",
121   ##    "\n", sep = "")
122   ##cat("Testing median differences...", "\n")
123   ##print(wilcox.test(data$x[data$y == 0], data$x[data$y == 1]))
124
125   # The following must be done for the case when id.col==F
126   x <- data$x
127   y <- data$y
128
129   ##cat("Performing Siegel-Tukey rank transformation...", "\n",
130   ##    "\n")
131
132
133
134   sort.x <- sort(data$x)
135   sort.id <- data$y[order(data$x)]
136
137   data.matrix <- data.frame(sort.x, sort.id)
138
139   base1 <- c(1, 4)
140   iterator1 <- matrix(seq(from = 1, to = length(x), by = 4)) -
141     1
142   rank1 <- apply(iterator1, 1, function(x) x + base1)
143
144   iterator2 <- matrix(seq(from = 2, to = length(x), by = 4))
145   base2 <- c(0, 1)
146   rank2 <- apply(iterator2, 1, function(x) x + base2)
147
148   #print(rank1)
149   #print(rank2)
150
151   if (length(rank1) == length(rank2)) {
152     rank <- c(rank1[1:floor(length(x)/2)], rev(rank2[1:ceiling(length(x)/2)]))
153   } else {
154     rank <- c(rank1[1:ceiling(length(x)/2)], rev(rank2[1:floor(length(x)/2)]))
155   }
156
157
158   unique.ranks <- tapply(rank, sort.x, mean)
159   unique.x <- as.numeric(as.character(names(unique.ranks)))
160
161   rank.matrix <- data.frame(unique.x, unique.ranks)
162
163   ST.matrix <- merge(data.matrix, rank.matrix, by.x = "sort.x",
164                      by.y = "unique.x")
165
166   ##print(ST.matrix)
167
168   ##cat("\n", "Performing Siegel-Tukey test...", "\n", sep = "")
169
170   ranks0 <- ST.matrix$unique.ranks[ST.matrix$sort.id == 0]
171   ranks1 <- ST.matrix$unique.ranks[ST.matrix$sort.id == 1]
172
173   ##cat("\n", "Mean rank of group 0: ", mean(ranks0), "\n", sep = "")
174   ##cat("Mean rank of group 1: ", mean(ranks1), "\n", sep = "")
175
176   ##print(wilcox.test(ranks0, ranks1, alternative = alternative,
177   ##                  mu = mu, paired = paired, exact = exact, correct = correct,
178   ##                  conf.int = conf.int, conf.level = conf.level))
179
180   #putting function outputs as objects so can be extracted
181   mean.rank0 <- mean(ranks0)
182   mean.rank1 <- mean(ranks1)
183   st.test <- wilcox.test(ranks0, ranks1, alternative = alternative,
184                          mu = mu, paired = paired, exact = exact, correct = correct
```

```
                                ,
185                         conf.int = conf.int, conf.level = conf.level)
186
187   return(list('st.ranks'=ST.matrix, 'mean.rank0'=mean.rank0,
188               'mean.rank1'=mean.rank1, 'st.test'=st.test ))
189 }
190
191
192
193
194
195
196
197 if(F) {
198
199   #Example:
200
201   ### 1
202   x=c(4,4,5,5,6,6)
203   y=c(0,0,1,9,10,10)
204   siegel.tukey(x,y, F)
205   siegel.tukey(x,y) #same as above
206
207   ### 2
208   # example for a non equal number of cases:
209   x=c(4,4,5,5,6,6)
210   y=c(0,0,1,9,10)
211   siegel.tukey(x,y,F)
212
213   ### 3
214   x <- c(33, 62, 84, 85, 88, 93, 97, 4, 16, 48, 51, 66, 98)
215   id <- c(0,0,0,0,0,0,0,1,1,1,1,1,1)
216   siegel.tukey(x,id,T)
217   siegel.tukey(x~id) # from now on, this also works as a function...
218   siegel.tukey(x,id,T,adjust.median=F,exact=T)
219
220   ### 4
221   x<-c
          (177,200,227,230,232,268,272,297,47,105,126,142,158,172,197,220,225,230,262,270)

222   id<-c(rep(0,8),rep(1,12))
223   siegel.tukey(x,id,T,adjust.median=T)
224
225
226   ### 5
227   x=c(33,62,84,85,88,93,97)
228   y=c(4,16,48,51,66,98)
229   siegel.tukey(x,y)
230
231   ### 6
232   x<-c(0,0,1,4,4,5,5,6,6,9,10,10)
233   id<-c(0,0,0,1,1,1,1,1,1,0,0,0)
234   siegel.tukey(x,id,T)
235
236   ### 7
237   x <- c(85,106,96, 105, 104, 108, 86)
238   id<-c(0,0,1,1,1,1,1)
239   siegel.tukey(x,id,T)
240
241 }
```

stat custom.R

```
1 #////////////////////////////////////////////////////////////////////////////
      //
2 #Function to get descriptive statistices
3 #meant to be used inside a ddply
4
```

```r
#getting mean normalized values for each compound
stat_custom <- function(d, value) {
  stat <- data.frame(avg = mean(d[,value]),
                     med = median(d[,value][d[,value] != 0]),
                     stdev = sd(d[,value]),
                     n=length(d[,value]))

  stat$se <- stat$stdev/sqrt(stat$n)
  stat$ymax <- stat$avg + stat$se
  stat$ymin <- stat$avg - stat$se

  stat$r_max <- range(d[,value])[2]
  stat$r_min <- range(d[,value])[1]
  stat$per_r_max <- abs(stat$r_max-stat$avg)/stat$avg*100
  stat$per_r_min <- abs(stat$r_min-stat$avg)/stat$avg*100

  out <- cbind(d,stat)
  return(out)
}
```

linear model.R

```r
#////////////////////////////////////////////////////////////////////////////////
#Test for linear model

#Extracts estimate, stanadard error, test statistic, p-value for slope and
    intercept

lm_custom <- function(data, group, value='value'){
  d <- data
  f <- paste(value, group, sep='~')

  result <- lm(as.formula(f), data=d)
  result.sum <- summary(result)

  out <- data.frame(slope=result.sum$coefficients[2,1],
                    slope.se=result.sum$coefficients[2,2],
                    slope.t=result.sum$coefficients[2,3],
                    slope.p=result.sum$coefficients[2,4],
                    int=result.sum$coefficients[1,1],
                    int.se=result.sum$coefficients[1,2],
                    int.t=result.sum$coefficients[1,3],
                    int.p=result.sum$coefficients[1,4])

  return(out)
}
```

normalize.R

```r
#////////////////////////////////////////////////////////////////////////////
#Function to normalize concentrations by one compound
#written for Fecal water samples

norm_custom <- function(d.stat, variable='variable',value='value',
                        norm='Acetate', norm.val='avg',norm.name='norm') {
  norm.value <- unique(d.stat[,norm.val][d.stat[,variable]==norm])
  d.stat[,norm.name] <- d.stat[,value]/norm.value
  out <- unique(d.stat[,!colnames(d.stat) %in%
                         c('med','stdev','n','se','ymax','ymin')])
  return(out)
}
```

multiple linear regression.R

```r
#////////////////////////////////////////////////////////////////////////////
#multiple linear regression
```

```r
#for comparison of bias and interaction between more than one independent variables

mlr_custom <- function(d, f){
  l.model <- lm(formula=f, data=d)
  l.result <- summary(l.model)
  l.coef <- as.data.frame(l.result$coefficient)
  rname <- data.frame(rname=rownames(l.coef))
  out <- cbind(rname, l.coef)
  out$variable <- as.character(unique(d$variable))
  return(out)
}
```

mlr prediction.R

```r
#////////////////////////////////////////////////////////////////////////////
#Predicting data points along the plane of a multiple linear regression
#obtains mlr model first, then applies data points based on that model
mlr_predict <- function(d, f){
  l.model <- lm(formula=f, data=d)
  d$prediction <- predict(l.model)
  return(d)
}
```

linear model.R

```r
#////////////////////////////////////////////////////////////////////////////
/
#Test for linear model

#Extracts estimate, stanadard error, test statistic, p-value for slope and
    intercept

lm_custom <- function(data, group){
  d <- data
  f <- paste('value', group, sep='~')

  result <- lm(as.formula(f), data=d)
  result.sum <- summary(result)

  out <- data.frame(slope=result.sum$coefficients[2,1],
                    slope.se=result.sum$coefficients[2,2],
                    slope.t=result.sum$coefficients[2,3],
                    slope.p=result.sum$coefficients[2,4],
                    int=result.sum$coefficients[1,1],
                    int.se=result.sum$coefficients[1,2],
                    int.t=result.sum$coefficients[1,3],
                    int.p=result.sum$coefficients[1,4],
                    res.sum=sum(result.sum$residuals))

  return(out)
}
```

ANOVA-mean.R

```r
#////////////////////////////////////////////////////////////////////////////
//

#ANOVA
#Testing variation about the mean within and between groups
aov_custom <- function(data, group) {
  d <- data

  grand <- mean(d$value) #grand mean

  ss.grp <- ddply(d, group, mutate, avg=mean(value)) #within-group mean
  ss.grp$grand <- grand #adding grand mean

```

```
  ssr <- sum((ss.grp$value - ss.grp$avg)**2) #sum of sqr of within groups
  ssa <- sum((ss.grp$avg - ss.grp$grand)**2) #sum of sqr of between groups

  dfr <- nrow(d)-length(unique(d[,group])) #n-q
  dfa <- length(unique(d[,group]))-1 #q-1

  msa <- ssa/dfa #mean square between group
  msr <- ssr/dfr #mean square within group

  f <- msa/msr #f statistic
  p.val <- 1-pf(f, dfa, dfr)
  out <- data.frame(variable=unique(d$variable), f.val=f, p.val=p.val)
  return(out)
}
```

kruskal wallis.R

```
#////////////////////////////////////////////////////////////////////////////////
    /
#kruskal-Wallis for UC expt 2

#kruskal wallis test
kruskal_custom <- function(data, group){
  d <- data
  result <- kruskal.test(data$value, as.factor(data[,group]))$p.val
  d$p.val <- result
  return(d)
}
```

tukey.R

```
#////////////////////////////////////////////////////////////////////////////////
    //
#Tukey's HSD function for UC Expt 2

tukey_custom <- function(data, group) {
  grps <- unique(data[,group])
  pair <- combn(grps,2)
  out <- c()
  for(i in 1:ncol(pair)) {
    grp1 <- data[data[,group] == pair[1,i],]
    grp2 <- data[data[,group] == pair[2,i],]

    grand <- mean(data$value) #grand mean
    ss.grp <- ddply(data, group, mutate, avg=mean(value)) #within-group mean
    ss.grp$grand <- grand #adding grand mean
    ssr <- sum((ss.grp$value - ss.grp$avg)**2) #sum of sqr of within groups
    dfr <- nrow(data)-length(unique(data[,group])) #n-q

    avg1 <- mean(grp1$value)
    avg2 <- mean(grp2$value)
    n <- nrow(grp1)
    se <- sqrt(ssr/n)

    q.stat <- (max(c(avg1, avg2))-min(c(avg1, avg2)))/se
    p.val <- ptukey(q.stat, n, df=dfr)

    output <- data.frame(g1=pair[1,i], g2=pair[2,i], p.val=p.val)
    out <- rbind(out, output)
  }
  return(out)
}
```

correlation.R

```
#////////////////////////////////////////////////////////////////////////////////
    //
```

```
#Correlation relationship

#Returns correlation coefficient, p-value and test statistic


corr_custom <- function(data, x, y, methods='pearson') {
  d <- data
  x <- as.numeric(data[,x])
  y <- as.numeric(data[,y])
  corr.result <- cor.test(x, y, method=methods, alternative='two.sided')
  p.val <- corr.result$p.value
  stat <- corr.result$statistic
  estimate <- corr.result$estimate
  out <- data.frame(stat=stat, estimate=estimate, p.val=p.val)
  return(out)
}
```

Student t test.R

```
#////////////////////////////////////////////////////////////////////////////////
    //
#Function to perform Student's t test

t_custom <- function(data, group) {

  grps <- unique(data[,group])
  grp1 <- data[data[,group] == grps[1],]
  grp2 <- data[data[,group] == grps[2],]

  result <- t.test(grp1$value, grp2$value)
  t.stat <- result$statistic
  p.val <- result$p.value

  out <- c(variable=unique(data$variable), t.stat=t.stat, p.val=p.val)

  return(out)
}
```

function-heat map bin.R

```
#////////////////////////////////////////////////////////////////////////////////
    //////////////
#Function to divide data into bins, and have them returned in a particular order
# best to be used for plotting heat maps
# only makes 5 bins

#input: data = dataframe that contains the value to be represented in plot; long
    form
#       bin = vector of 5 values are the bin limits, in decreasing order
#       variable = name of column that holds the variables to be represented in the
     plot
#       order.data = dataframe that holds the values by which the compounds should
    be ordered;
#                    usually is the same as data
#       order.value = name of column that holds values by which order.var should be
     ordered
#       order.var = name of column the variables to be ordered;
#                   order.var and variable should hold the same info

#output: uni.list = class list; list of compounds in each bin
#        ordered.data = dataframe of data to be plotted, and compounds are ordered
#                       by a decreasing value, usually mean concentration
#-------------------------------------------------------------------------------

bin.heatmap <- function(data, bin, variable, order.data, order.var, order.value) {


```

```
  #Assigning data to bins
  order.data$bin <- NA
  order.data[order.data[,order.value] >= bin[1], 'bin'] <- 'bin1'
  order.data[order.data[,order.value] >= bin[2] & order.data[,order.value] < bin
      [1], 'bin'] <- 'bin2'
  order.data[order.data[,order.value] >= bin[3] & order.data[,order.value] < bin
      [2], 'bin'] <- 'bin3'
  order.data[order.data[,order.value] >= bin[4] & order.data[,order.value] < bin
      [3], 'bin'] <- 'bin4'
  order.data[order.data[,order.value] >= bin[5] & order.data[,order.value] < bin
      [4], 'bin'] <- 'bin5'

  #converting compounds to character class and also gettinng compounds in each bin
  bin.cmpd <- list()
  for(i in c('bin1','bin2','bin3','bin4','bin5')) {
    temp <- order.data[order.data$bin == i,]
    bin.cmpd[[i]] <- as.character(unique(temp[,variable]))
  }

  #Getting only compounds found in one bin and not any other bins before it
  common.cmpd <- c(bin.cmpd[[1]])
  uni.list <- list(bin.cmpd[[1]])

  for(i in 2:length(bin.cmpd)) {
    uni.list[[i]] <- bin.cmpd[[i]][!bin.cmpd[[i]] %in% common.cmpd]
    common.cmpd <- c(common.cmpd, uni.list[[i]])
  }

  #within bins, ordering compounds by order.value and setting variable order
  ordered.data <- c()
  ordered.list <- list()
  for(i in 1:length(uni.list)) {
    cmpd <- uni.list[[i]]
    temp.data <- order.data[order.data[,order.var] %in% cmpd,]
    colnames(temp.data)[colnames(temp.data)==order.value] <- 'order.val'
    max.data <- ddply(temp.data, order.var, summarize,
                      max.value=max(order.val))
    temp.order <- max.data[order(max.data$max.value, decreasing=TRUE),]
    temp.order[,order.var] <- factor(temp.order[,order.var],
                                     levels=rev(temp.order[,order.var]),
                                     ordered=TRUE)

    temp <- data[data[,variable] %in% cmpd,]
    temp[,variable] <- factor(temp[,variable],  levels=levels(temp.order[,order.var
        ]))
    ordered.data <- rbind(ordered.data, temp)

    temp.list <- unique(temp[,variable])
    temp.list <- temp.list[order(temp.list, decreasing=TRUE)]
    ordered.list[[i]] <- as.character(temp.list)
  }

  return(list('uni.list'=ordered.list, 'ordered.data'=ordered.data))

}
```

function-bin function.R

```
#////////////////////////////////////////////////////////////////////////////////
    ///////
#Function to bin data

#input: data = dataframe that contains the value to be represented in plot; long
    form
#       bin = vector of 5 values are the bin limits, in decreasing order
#       variable = name of column that holds the variables to be binned
#       value = name of column that holds values that determines the bins

```

```r
#output: uni.list = class list; list of compounds in each bin
#-----------------------------------------------------------------------------
bin_function <- function(data, bin, variable='variable', value='value') {

  #Assigning data to bins
  data$bin <- NA
  data[data[,value] >= bin[1], 'bin'] <- 'bin1'
  data[data[,value] >= bin[2] & data[,value] < bin[1], 'bin'] <- 'bin2'
  data[data[,value] >= bin[3] & data[,value] < bin[2], 'bin'] <- 'bin3'
  data[data[,value] >= bin[4] & data[,value] < bin[3], 'bin'] <- 'bin4'
  data[data[,value] >= bin[5] & data[,value] < bin[4], 'bin'] <- 'bin5'

  #converting compounds to character class and also gettinng compounds in each bin
  bin.cmpd <- list()
  for(i in c('bin1','bin2','bin3','bin4','bin5')) {
    temp <- data[data$bin == i,]
    bin.cmpd[[i]] <- as.character(unique(temp[,variable]))
  }

  #Getting only compounds found in one bin and not any other bins before it
  common.cmpd <- c(bin.cmpd[[1]])
  uni.list <- list(bin.cmpd[[1]])

  for(i in 2:length(bin.cmpd)) {
    uni.list[[i]] <- bin.cmpd[[i]][!bin.cmpd[[i]] %in% common.cmpd]
    common.cmpd <- c(common.cmpd, uni.list[[i]])
  }

  return(uni.list)
}
```

defined binning.R

```r
#////////////////////////////////////////////////////////////////////////////
#Modified for Liquid Gold Manuscript A
#written for Manuscript A -- function to define bins, and return which bin
#each compound is in.

defined_binning <- function() {

  #Setting working directory
  setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Analysis')

  #loading required packages
  require(RSQLite)
  require(ggplot2)
  require(plyr)
  require(reshape2)
  require(gridExtra)

  #sourcing required functions
  source('../Functions/function-data cleanup.R')
  source('../Functions/function-descriptive statistics.R')

  source('../Functions/function-heat map bin.R')


  #Input data
  #-----------------------------------------------------------------------------

  # Choose driver
  drv <- dbDriver('SQLite')

  # Create connection (for SQLite, this is a file)
  con <- dbConnect(drv, '../Data source files/data.db')

  input <- dbGetQuery(con, "SELECT q.sampleID, q.donor, q.run, q.vessel, q.DPI,
                      q.hour, q.phase, q.dose, q.treatment,q.gen_treat,
```

```
                        q.expt, d.variable, d.value
                        FROM Data d INNER JOIN Qualifier q
                        ON d.file = q.file
                        INNER JOIN Rank r
                        ON d.variable = r.Compound AND q.expt = r.expt
                        WHERE r.Confidence = 1 AND q.gen_treat = 'None'
                        AND d.variable NOT IN ('Xanthine','Glycerol')
                        AND q.state = 'ss'
                        AND (q.run != 8 OR q.run IS NULL)
                        AND q.donor != 'RP1'
                        AND q.expt != 'RP2'
                        ORDER BY q.sampleID;")

cmpd.info <- dbGetQuery(con, "SELECT c.Compound, c.Type, p.Pathway, m.main
                            FROM Cmpd_type c INNER JOIN Pathway p
                            ON c.Compound = p.Compound
                            INNER JOIN Main_pathway m
                            ON p.Pathway = m.path
                            ORDER BY c.Compound")

dbDisconnect(con)

#Setting working directory
setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/LG Manuscript A/Analysis')

#data cleanup
-------------------------------------------------------------------------------



#Performing data cleanup to remove zero value compounds
#also casts data into wide format
clean <- data_cleanup(input, form.y=c('sampleID','donor','run','vessel',
                                      'DPI','hour','phase','dose','treatment',
                                      'gen_treat', 'expt'),
                      form.x='variable', convert.NA=TRUE)

#make sure all compounds have a value for each donor
data.cast <- dcast(clean, formula=sampleID+donor+run+vessel+DPI+hour+phase+dose+
                   treatment+gen_treat+expt~variable, value.var='value')

data.cast[is.na(data.cast)] <- 0

#melting back to long format
data.melt <- melt(data.cast, id.vars=c('sampleID','donor','run','vessel',
                                       'DPI','hour','phase','dose','treatment',
                                       'gen_treat', 'expt'))


#descriptive stats
-------------------------------------------------------------------------------


#Getting descriptive stats for each donor
stat <- ddply(data.melt, 'donor', descriptive.stat, data.type='long', variable='variable',
              qualifier=qualifier, scale.data=FALSE)

#Assigning data to bins
bin.data <- bin.heatmap(stat, bin=c(20,10,1,0.1,0), 'variable',
                        order.data=stat, order.var='variable', order.value='avg')

uni.list <- bin.data[['uni.list']]

#getting dataframe of variables and their bins
bins <- c()
for(i in 1:length(uni.list)) {
  d <- data.frame(bin=i, variable=uni.list[[i]])
  bins <- rbind(bins, d)
```

```
  }

  return(bins)
}
```

function-weighted compound class.R

```
#////////////////////////////////////////////////////////////////////////////////
    /
#Function to illustrate profile diversity
#Getting proportion of compound types weighted to their concentrations
#to be used in ddply, grouped by donor

custom_perc <- function(d, cmpd.class='Type') {

  #First, getting sum of all compound concentrations for each sample
  output <- ddply(d, 'sampleID', mutate, sample.total=sum(value, na.rm=TRUE))

  #Getting mean profile totals
  output$sample.avg <- mean(output$sample.total)

  #Sum of compound classes within samples
  output <- ddply(output, c('sampleID',cmpd.class), mutate,
                  sample.class=sum(value, na.rm=TRUE))

  #Mean compound class
  output <- ddply(output, cmpd.class, mutate, class.avg=mean(sample.class))

  #Mean compound class as percent of profile total
  output$perc <- output$class.avg/output$sample.avg*100

  out <- unique(output[,c('donor','sample.avg',cmpd.class,'class.avg','perc')])
  return(out)
}
```

function-defined binning.R

```
#////////////////////////////////////////////////////////////////////////////////
    //////
#Function to get order of control data - modified for thesis


#input: data = dataframe of non-treated samples;
#              holds the values by which the compounds should be ordered

#output: order.list = dataframe of data to be plotted, and compounds are ordered
#                     by a decreasing value
#-------------------------------------------------------------------------------

defined_binning <- function() {

  wd <- getwd()

  #Setting working directory
  setwd('C:/Users/Sandi/Dropbox/Projects/Liquid Gold/R Scripts/V 3.0/Analysis')

  #loading required packages
  require(RSQLite)
  require(ggplot2)
  require(plyr)
  require(reshape2)
  require(gridExtra)

  #sourcing required functions
  source('../Functions/function-data cleanup.R')
  source('../Functions/function-descriptive statistics.R')
  source('../Functions/function-heat map bin.R')
```

```
30
31
32  #Input data
    ------------------------------------------------------------------------------

33  # Create connection and selecting database
34  con <-  odbcConnect('MySQLDSN')
35
36  input <- sqlQuery(con, "SELECT q.sampleID, q.donor, q.run, q.vessel, q.DPI,
37                     q.hour, q.phase, q.dose, q.treatment,q.gentreat, q.expt,
38                     d.variable, d.value, t.tax_class
39                     FROM liquidgold.data d INNER JOIN liquidgold.qualifier q
40                     ON d.file = q.file
41                     INNER JOIN liquidgold.rank r
42                     ON d.variable = r.Compound AND q.expt = r.expt
43                     INNER JOIN general.tax t
44                     ON d.variable=t.syn
45                     WHERE r.Confidence = 1 AND q.gentreat = 'None'
46                     AND (q.run != 8 OR q.run IS NULL)
47                     AND d.variable NOT IN ('Xanthine','Glycerol')
48                     AND q.state = 'ss'
49                     AND q.donor != 'RP1'
50                     AND q.expt != 'RP2'
51                     ORDER BY q.sampleID;")
52
53  odbcClose(con)
54
55  #data cleanup
    ------------------------------------------------------------------------------


56
57  #Performing data cleanup to remove zero value compounds
58  #also casts data into wide format
59  clean <- data_cleanup(input, convert.NA=TRUE)
60
61  #make sure all compounds have a value for each donor
62  data.cast <- dcast(clean, formula=sampleID+donor+run+vessel+DPI+hour+phase+dose+
63                       treatment+gentreat+expt~variable, value.var='value')
64
65  data.cast[,12:ncol(data.cast)][is.na(data.cast[,12:ncol(data.cast)])] <- 0
66
67  #melting back to long format
68  data.melt <- melt(data.cast, id.vars=c('sampleID','donor','run','vessel',
69                                         'DPI','hour','phase','dose','treatment',
70                                         'gentreat', 'expt'))
71
72  data.melt <- unique(merge(data.melt, clean[,c('tax_class','variable')],
73                            by='variable'))
74  #separating out qualifier and data
75  qualifier <- data.melt[,1:11]
76  x.data <- data.melt[,12:13]
77
78  #descriptive stats
    ------------------------------------------------------------------------------

79  #Getting descriptive stats for each donor
80  stat <- ddply(data.melt, 'donor', descriptive.stat, data.type='long',
81                variable='variable')
82
83  #Binning
    ------------------------------------------------------------------------------

84  #Assigning data to bins
85  bin.data <- bin.heatmap(stat, bin=c(20,10,1,0.1,0), 'variable',
86                          order.data=stat, order.var='variable', order.value='avg')
87
88  uni.list <- bin.data[['uni.list']]
89  ordered.data <- bin.data[['ordered.data']]
```

```
90
91    var <- levels(ordered.data$variable)
92    var <- factor(var, levels=var, ordered=TRUE)
93
94    for(i in 1:length(uni.list)){
95      uni.list[[i]] <- factor(uni.list[[i]], levels=var)
96    }
97
98    #setting working directory back
99    setwd <- wd
100   return(uni.list)
101 }
```